

ATARI AT DISNEYLAND

IPC 90078



THE ATARI ST MONTHLY MAGAZINE

SEPTEMBER 1989

ISSUE 36

DISK VERSION \$12.95

MAGNIWRITER ST EXPERIMENT 42 THE ANIMATION STAND

REVIEWS:

I.B. Dual Drives
The Atari ST Book
Falcon
Bubble Bobble
Manhunter: New York



09

0 74369 50078 5

ATARI

53124



The NEXT Generation ★★

"There is no question that this version of Calamus blows its Atari competitors out of the water..."

"In fact, Calamus, in my opinion, competes neck-and-neck with all page layout programs on the Mac and the IBM."

Personal Publishing, September 1988

At a suggested retail price of US \$299.95, Calamus is setting a new price performance benchmark for the desktop publishing industry.

For more information or to place your order, call or write to:
ISD Marketing, Inc. 2651 John Street, Unit #3 Markham, Ontario, L3R-2W5, Canada.

Tel. 416 479-1880, Fax: 416 479-1882.

All computers and software are tradenames and/or trademarks of their respective manufacturers.

R

ecently, people complained that it was difficult to determine whether a program in SFLOG was "disk-only" or "type-in." Originally, we included this information at the beginning of a program's accompanying article, but many felt that they should not have to flip through the magazine in order to figure out exactly what they were buying. We agreed. So for the last couple of issues we have included the "disk-only" information in each program's description in the table of contents. This method, although effective, is anything but elegant.

Finally, it was suggested by D.A. Brumleve, the author of SFLOG's children's column, *The CompuKid Connection*, that what we really need is an icon that can convey, at a glance, the necessary information about each program. She further suggested that this icon indicate not only whether a program is disk-only or type-in, but also whether it runs in color, monochrome or both.

I thought this was a great idea, but also felt that a single icon that could convey this much information would be difficult to come up with. Ms. Brumleve disagreed, and to prove the point, she had ready, less than 24 hours later, exactly the type of icon she had described. Needless to say, we purchased it for use in SFLOG.

The bottom line is that, from now on, the SFLOG table of contents will contain icons that will give you, at a glance, the answers to the following questions:

- 1) Does the article offer a program on disk?
- 2) Is there a complete type-in listing?
- 3) Does the program run in color, monochrome or both?

Note the words "complete type-in listing" in question #2. We sometimes include, for the benefit of interested programmers, program listings that, although they show a complete program, cannot be run without subsidiary data files—usually picture files—that would be impossible to include in type-in form. We don't consider this type of program listing to be a type-in, so if an icon indicates a type-in listing, you may be sure that you do not require the disk to run the program.

At the bottom of the table of contents you'll find a key that will tell you how to interpret the icons. After reading the key once, you should never have to refer to it again. The icons are almost self-explanatory. From this point on, you'll never have to search through the magazine or read the entire table of contents to get the information you need about a particular program. I don't think we could possibly make it any simpler. You're welcome. ■



PHOTOGRAPHY: GARRY GAY

BY CLAYTON WALNUM



FEATURES

IN THIS ISSUE

- The Ultimate Screen Saver** Richard Leinecker 18
Prevent screen burn-in and enjoy a fireworks display at the same time
- The Animation Stand: There's the Windup** Maurice Molyneaux 32
The sixth installment of this tutorial on professional animation techniques
- Magnewriter ST** Bruce D. Noonan, M.D. 36
The popular ST Writer word processor is modified yet again, this time so that it's easier to use by those who are visually impaired.
- Experiment 42** Greg Knauss 42
Edwin, the mutant snake, has just found himself in a science lab where he's going to undergo experimentation. This revamping of a classic game concept will provide hours of slithering fun.
- The Michigan Atari Exposition** Derek Mihoicka 48
The M.A.C.E. show was a great success. Here's a brief look at what you missed if you didn't attend.
- Software Engineering:**
The Software Metric System Karl E. Wieggers 50
The series continues with a discussion of software-production evaluation techniques.
- Atari At Disneyland** Frank Cohen 58
A full report of the World of Alan Shaw held recently in Anaheim, California.
- RWABS Monitor** David Small & Dan Moore 70
The unique utility will let you watch your disk drive in action.

REVIEWS

- Turbo ST** David Plotkin 84
- The Atari ST Book** David Plotkin 85
- I.B. Dual Drives** Pamela Rice Hahn 86
- The ST Gameshell** 88
- This month Falcon, Bubble Bobble, Alcon ST, Space Station Oblivion, Technopop, Downhill Challenge and Manhunter: New York are reviewed by Steve Panak, Clayton Walnum, Frank Eva, Scott Wasser and Peter A. Smith.

COMPUKID CONNECTION



EXPERIMENT 42



SCREEN SAVER

COLUMNS

- PD Parade** George L. Smyth 7
- From Over the Big Water** Marshal M. Rosenthal 10
- C-ManShip** Clayton Walnum 14
- Step 1** Maurice Molyneaux 38
- Ian's Quest** Ian Chadwick 46
- ST User** Arthur Leyenberger 56
- The CompuKid Connection** D.A. Brumleve 65

DEPARTMENTS

- Editorial** Clayton Walnum 3
- ST News** 8
- ST Check** Clayton Walnum 82
- Disk Contents** 96
- Footnotes** Karl E. Wieggers 97

PROGRAM · LISTING · GUIDE

- SCREEN SAVER** page 20
- RWABS MONITOR** page 71
- C-MANSHIP** page 74

SYMBOL · GUIDE

- This program runs in both color and mono. It is available in type-in form & on the disk.
- This program runs in mono only. It is available in type-in form & on the disk.

- This program runs in mono only. It is available in type-in form & on the disk.
- This program runs in both color and mono. It is available only on the disk.
- (No image) This article does not have an associated program.

- This program runs in color only. It is available only on the disk.
- This program runs in mono only. It is available only on the disk.

ST-LOG (ISSN 0890-9601) is published monthly by L.F.P., Inc., 9171 Wilshire Blvd., Suite 300, Beverly Hills, CA 90210. © 1989 L.F.P., Inc. Return postage must accompany all manuscripts, drawings, photos, disks, etc., if they are to be returned, and no responsibility can be assumed for unsolicited materials. All rights reserved on entire contents; nothing may be reproduced in whole or in part without written permission of the publisher. U.S. subscription \$28 for one year (12 issues), \$52 for two years (24 issues), \$76 for three years (36 issues). Foreign subscriptions: add \$10 per subscription per year. Single copy \$3.50 (add \$1 for postage). Change of address: six weeks advance notice, and both old and new addresses are needed. POSTMASTER, send change of address to: ST-LOG Magazine, P.O. Box 16928, North Hollywood, CA 91615. Second-class postage paid at Beverly Hills, CA, and additional mailing offices.

SEPTEMBER 1989
ISSUE 35



STAFF

Publisher: Lee H. Pappas. **Executive Editor:** Clayton Walnum. **Associate Editor:** Andy Eddy.
Art Director: Andy Dean. **Managing Editor:** Dean Brierly. **East Coast Editor:** Arthur Leyenberger.
West Coast Editor: Charles F. Johnson. **Contributing Editors:** D.A. Brumleve, Ian Chadwick, Frank Cohen, Maurice Molyneux, Steve Pansk, Marshal M. Rosenthal, George L. Smyth. **Copy Chief:** Sarah Weinberg. **Copy Editors:** Alyson Gould, Sarah Bellum, Norma Edwards, Randolph Heard, Kim Turner. **Chief Typographer:** Alice Nichols. **Typographers:** David Buchanan, B. Miro Jr., Quia Saxon, Ligaya Rafael. **Editorial Assistant:** Patricia Koury. **Contributors:** Albert Baggetta, William Busby, Frank Eva, Bill Griggs, Kathy Schaefer, M.D., Ron Schaefer, M.D., Todd Threadgill, Scott Wasser. **Vice President, Production:** Donna Hahner. **Production Assistant:** Steve Hopkins.
National Advertising Director: Jay Eisenberg. **Corporate Director of Advertising:** Paula S. Thornton. **Advertising Production Director:** James Rosenblum. **Advertising Production Coordinator:** Maggie Chun. **Corporate Editorial:** Tim Conway. **Corporate Art:** Pam Carr.
Subscriptions Director: Irene Gradstein. **Vice President, Sales:** James Gustafson.

U.S. newsstand distribution by Eastern News Distributors, Inc., 1130 Cleveland Rd., Sandusky, OH 44870.

ST-LOG Magazine (L.F.P., Inc.) is in no way affiliated with Atari. Atari is a trademark of Atari Corp.

ADVERTISING SALES

Correspondence, letters and press releases should be sent to: Editor, **ST-LOG**, 9171 Wilshire Blvd., Suite 300, Beverly Hills, CA 90210.

Correspondence regarding subscriptions, including problems and changes of address, should be sent to: **ST-LOG**, P.O. Box 16928, North Hollywood, CA 91615 or call (818) 760-8983.

Correspondence concerning a regular column should be sent to our editorial address (see Authors below), with the name of the column included in the address. We cannot reply to all letters in these pages; so if you would like an answer, please enclose a self-addressed, stamped envelope.

J.E. Publishers Representatives — Los Angeles: (213) 467-2268.
6855 Santa Monica Blvd., Suite 200, Los Angeles, CA 90038.
San Francisco: (415) 864-3252. Chicago: (312) 445-2489.
Denver: (303) 595-4331.
New York: (212) 724-7767.

Address all advertising materials to: Paula Thornton — Advertising Director, **ST-LOG**, 9171 Wilshire Blvd., Suite 300, Beverly Hills, CA 90210.

PERMISSIONS

No portions of this magazine may be reproduced in any form without written permission from the publisher. Many of the programs printed herein are copyrighted and not public domain.

Due, however, to numerous requests from Atari club libraries and bulletin-board systems, our policy does allow club libraries or individually run BBSs to make certain programs from **ST-LOG** available during the month printed on that issue's cover. For example, software from the January issue can be made available January 1.

This does not apply to programs which specifically state that they are not public domain and, thus, are not for public distribution.

In addition, any programs used must state that they are taken from **ST-LOG** Magazine. For further information, contact **ST-LOG** at (203) 645-6236.

SUBSCRIPTIONS

ST-LOG, P.O. Box 16928, North Hollywood, CA 91615 or call (818) 760-8983. Payable in U.S. funds only. U.S.: \$28.00-1 year; \$52.00-2 years; \$76.00-3 years. Foreign add \$7.00 per year per subscription. For disk subscriptions, see the cards at the back of this issue.

AUTHORS

When submitting articles and programs, both program listings and text should be provided in printed and magnetic form, if possible. Typed or printed text copy is mandatory and should be in upper and lowercase, with double spacing. If a submission is to be returned, please send a self-addressed, stamped envelope with your manuscript to **ST-LOG**, P.O. Box 1413-MO, Manchester, CT 06040-1413.

COVER PHOTOGRAPHY:
LADI VON JANSKY

COVER MODEL:
LEIGH WALSH

INTERIOR PHOTOGRAPHY:
LOUIS CASTENEDA
GARY GAY
ELLEN SHUSTER

ILLUSTRATORS:
JOHN BERARD
WAYNE T. GLADYSZ
FABIAN MASON

THANKS TO:
IMAGE BANK WEST
WESTLIGHT

A LETTER FROM THE PUBLISHER

It's no secret that the U.S. Atari market isn't as healthy as it could be. The 8-bit computer line has declined in popularity, while the ST, though it has gained a respectable following in Europe, has yet to find its niche in the states. For these reasons, most software companies won't develop products for the Atari systems.

This lack of software support has a subtle, but nonetheless powerful impact on magazines that rely on the Atari market for their well-being. The cold fact is that advertisers for the 8-bit products are nearly nonexistent, and there are precious few advertisers for ST products.

Since, for profitable publications, we depend to a great extent upon advertising, we are left with two choices if our publications are to continue: We can increase the price of our magazines, thus forcing readers to pick up the tab for the lack of advertising, or we can find a way to make the magazines less expensive to produce. We've opted for the latter.

There are, of course, many ways we can cut the magazines' publishing costs: We can reduce the page count. We can get rid of the color. We can pay contributors less. Unfortunately, none of these options, nor others, not mentioned here, makes much of a difference in the long run.

After much thought, we decided that although the Atari market is not capable of supporting two Atari-specific magazines from a single publisher, it *is* active enough to support one. So we're going to combine ANALOG Computing and ST-LOG into a single monthly publication.

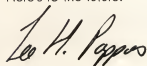
Don't panic! When you think about it, the merging of the magazines will allow us to produce a much nicer publication. And since the single magazine will be larger than either of the individual ones, we won't have to cut much from our content. In fact, after doing some analysis, we've discovered that we will be able to offer the same columns, departments and types of features you've come to expect. Little will change, except that everything will come to you under a single cover.

The November issue will be the first combination magazine. Next month we'll give you more details on what the new publication will be like, as well as our plans for the future. (We plan some nice surprises, like a reduction in the cost of magazine disks.)

We believe that merging ANALOG Computing and ST-LOG is the best solution to a tough problem. It allows us to continue publication while giving you your full money's worth. It also gives Atari a chance to prove their claim that in the coming year they will emerge a strong presence in the U.S. When that time comes, we plan to reevaluate the situation and possibly separate the publications once again.

Recently, Atari supporters have had to stick together like never before. We've been there, providing support and information for nearly nine years. And we plan to be there for many more.

Here's to the future!



Lee H. Pappas
Publisher



The program described here is available on this month's disk version, from the databases of the ST-LOG ST users' group on DELPHI, and from various BBSs and online networks across the U.S.

Last month I completed the task of relocating to a more favorable area. In spite of the fact that the move only consisted of a ten-mile displacement and was facilitated by multiple trips with a rented truck, the inevitable series of misfortunes associated with this chore accompanied my efforts.

The most embarrassing of these problems involved the misplacement of the small pieces of paper bearing the names and phone numbers of my friends. As some have unlisted phone numbers, I was not able to contact them until one of the last boxes was opened. Being one to give my computer access to nearly everything, I resolved to enter these scraps of information in an electronic address book.

The program I chose, *Address Base 0.9* by Steve MacMillan, provides all the essential elements found in an electronic address book. This medium-resolution program has the ability to load, save and merge files; add, edit and delete records and print the entered information.

When the program is invoked from the desktop, a GEM window displays the information and options. The menu bar holds the options Desk, Files, Options, Print and Dial. The letters A-Z are displayed on the right side of the screen, with two options labeled SEARCH and ADD below them.

A record is added to the file by either choosing the Add Record line in the Options portion of the menu bar or by pressing the control key and the letter "A". This places the cursor in the first field of the entry screen, which includes prompts for Last Name, First Name, Address, City, State, Zip, Phone #1, Phone #2, Attn, Names and Subgroup entries.

When editing the Up and Down arrow keys move the cursor from one field to another, as the Right and Left arrow keys move the cursor from one character to the next. Pressing the undo key clears the line and places the cursor in the first position of the field. Choosing the Add Record selection or pressing the control key and the letter "A" adds the record and returns control to the mouse.

A record is edited by displaying the record and choosing the Edit Record option from the menu bar or by pressing Control-E. Another method is to simply click on the field to be edited. After

PD PARADE

BY GEORGE L. SMYTH

Desk Files Options Print Dial	
ADDRESS BASE PROGRAM	
Last Name: Rogers	A B
First Name: Henry	C D
Address: 563 North Main Street	E F
City: Greensboro	G H
State: NC	I J
Zip: 43764	K L
Phone #1: (999) 645-8645	M N
Phone #2: (999) 645-8759	O P
Attn:	Q R
Notes: Wife's name: Margaret	S T
	U V
	W X
	Y Z
	SEARCH
	ADD
Sub Group: _____	

modification, pressing Control-A updates the record.

Deleting a record is performed much like the above: choose the Delete Record option from the menu bar or press Control-D after displaying the affected record. The user is given a chance to back out in case the wrong key was pressed.

The records may be displayed via a variety of methods. The easiest and fastest way is to enter the "*" character. This function, called Fast Display, leaves the GEM window and displays the current record. The arrow keys are then used to page through the different entries. Pressing the "*" character again returns the user to the GEM window.

Within the window, records may be paged by using the scroll bar at the bottom of the screen. Clicking on the arrow portion of the bar moves the records one at a time, while moving the scroll bar allows a jump to any position within the list. Pressing the home key transfers the screen to the first record, while pressing the shift and home keys relocates the pointer to the last record.

A letter to the right of the screen may also be clicked. This action displays the first record, with the last name beginning with the chosen letter. I found this method to be the quickest way to find a name.

The Search option scans the address base for an entered non-case-sensitive string within a chosen field. As the Notes field has room for ten lines of entry, the Search feature can even have, to a small extent, database implications, as this field is available to selection criteria. When the search process is complete, the first rec-

ord found is displayed, with subsequent choices available by pressing Control-N. This search list can be printed or saved to disk as a separate file.

The Print option on the menu bar allows three choices: print all, print one and print search. If the "Phone # Only" switch is set, only the names and numbers will be printed. If this switch has not been activated, the printout is not formatted and merely dumps the information to the printer "as is."

Existing files may be merged into the file in memory by choosing the Merge option within the Files section of the menu bar. As long as the combined number of records does not exceed the maximum of 400, the files are combined and stored.

Loading and saving information is accomplished through GEM file selectors. The Files menu bar option also allows the user to alter the pathname of the saved files.

I hope I won't have to move again, but if I do, I will merely have to locate one disk to get my address book, as opposed to finding several dozen small pieces of paper. ■



George L. Smyth holds a degree in psychology from West Virginia University and is currently employed as a programmer. He is the author of a series of tutorials on programming in Fort.

More from Cinemaware

Cinemaware recently announced four new games for the Atari ST computers. The games—*Deathbringer*, *Federation*, *Total Eclipse* and *Dark Side*—are part of a new software line named Spotlight Software, which features top games from around the world.

According to Cinemaware's president Bob Jacob, "We are a rapidly expanding entertainment software company, and it is vital that we successfully compete in every entertainment category." He goes on to say that "Spotlight Software does just as its name suggests; it brings to the spotlight the very best game products we can find in the most popular game genres. With the addition of Spotlight Software, people can look to Cinemaware not only for the best interactive movies, but also for the finest fantasy role-playing, adventure, action-strategy, simulation and sports games."

Also just shipping from Cinemaware is *Speedball*, the first title from their new affiliate, Image Works.

Deathbringer, *Total Eclipse*, *Speedball* and *Dark Side* are priced at \$39.95. *Federation* is slightly higher at \$49.95.

Cinemaware

4165 Thousand Oaks Blvd.
Westlake Village, CA 91362
(805) 495-6515

CIRCLE #113 ON READER SERVICE CARD.



SPEEDBALL

According to Cinemaware's president Bob Jacob, "We are a rapidly expanding entertainment software company, and it is vital that we successfully compete in every entertainment category."



Mouse keyboard

If you've become tired of moving your hand from the mouse to the keyboard and back again, SoundSight Corporation has a product that may be of interest. FastKeys is a self-stick strip that adheres to the side of the mouse, providing ST owners the ability to trigger keys on the keyboard from the mouse. Programmable keys allow the setup of macros for even more flexibility. The suggested retail price is \$39.95.

SoundSight Corporation

2105 Acyona Drive · Los Angeles, CA 90068
(213) 463-9464

CIRCLE #115 ON READER SERVICE CARD.



Demon's Winter for the ST

Now available for the Atari ST is SSI's (Strategic Simulations) fantasy-adventure game *Demon's Winter*, in which your five-character party may consist of dwarves, elves or dark elves. Each character has his own speed, strength, intellect, endurance

In *Demon's Winter* each character has his own speed, strength, intellect, endurance and skill levels. Each of these attributes can be increased by collecting experience points.

ance and skill levels. Each of these attributes can be increased by collecting experience points. Characters can be from one of many different classes, including barbarian, thief, wizard and scholar.

Demon's Winter is distributed by Electronic Arts under its Affiliated Labels program and is priced at \$39.95.

Strategic Simulations, Inc.
1046 N. Rengstorff Avenue
Mountain View, CA 94043
(415) 964-1353

CIRCLE #115 ON READER SERVICE CARD.

DEMON'S WINTER

FROM OVER THE BIG WATER

By Marshal M. Rosenthal

There's an association between companies and products. Band-Aid, Xerox, they've become generic names. It hasn't quite reached that state with software, but a few names always stick out: Activision (now Mediagenic), Electronic Arts, Sierra On-line. It's that way overseas, too, and the name in Britain that falls into this category is U.S. Gold.

The "big G" used to import software from the States to distribute in England. Now that things have turned around, with stuff coming from there to here, U.S. Gold has also shifted focus slightly, bringing in software from all over Europe to the U.S. (one source here is Epyx). The result is that a lot of great ST stuff comes through their doors. Here's a selection to keep you occupied.

First up is *Road Blasters*. This conversion from the Atari coin-op is not only faithful to the original, it also has graphics and sound that nearly match. Your super-charged auto barrels down the road, scoring points by shooting vehicles in the way. Sort of *Pole Position* with a vengeance, right? Start off with a relatively simple machine blaster, but plan on upgrading to greater firepower. You see, there's this jet that circles overhead and drops packages to pick up—packages containing anything from an Uzi cannon to extra fuel to a trio of Cruise missiles.

Fuel is the big problem here. The main tank empties quickly if you sit there idling, so move! Fortunately there are those green fuel balls to run over, but you have to avoid the toxic spills and mines at the same time. Meanwhile, there are darting Rat Jeeps and annoyingly small motorcycles getting in the way. So it's one eye on the road, one eye on the instrument panel, one eye scanning the skies, one eye....

Quick wits and reflexes are the rule, as each weapon has limited energy (with a power graph to monitor). The fire button is used for shooting, while pressing the button and pulling back on the stick fires a Cruise missile (should you be so lucky as to get some). These babies take out everything ahead of you, including those heavily armored Command cars. Completing a course moves you to a higher level, and there are three areas in which to race (each having its own distinct background, like Bubble City). There's a menu screen that offers choices, such as music on/off and whether to play with joystick or mouse. (Joystick, fer shurr.)

Did I forget to mention that leaving the road could result in your destruction? Don't



ROAD BLASTERS

This conversion from the Atari coin-op is not only faithful to the original, but it also has graphics and sound that nearly match. Your supercharged auto barrels down the road, scoring points by shooting vehicles in the way.

try hitting the poles or trees to find out what happens. There are also automated gun turrets that fire as you pass. Revving up the engine sends you off in a squeal of tires and a cloud of smoke, perhaps the best set of effects in the whole game. Remember to fasten your seat belt!



F R O M • U . S . G O L D

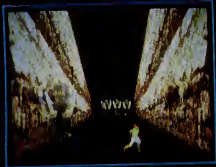
This is an action game that delivers. Moving laterally "forces" the screen from side to side. You can't always see what you're up against until it's on top of you.

The hideous Baculacites have kidnapped Princess Sheeta. Time to grab a car and get after her. *Last Duel* is definitely a shoot-'em-up. The vertically moving landscape doesn't stop for your car, and the various baddies keep shooting at you regardless of what you do. Of course, what to do is keep moving, shooting them first, and picking up extra power and time by blasting the special icons. Last long enough and you reach the end of the road—meaning a bad, bad, baddie. I've yet to last that long, but should you survive, a "teaser" screen appears as Princess Sheeta begs for deliverance.

Animation is fast, though the graphics are a bit crude. But who cares? This is an action game that delivers. Moving laterally "forces" the screen from side to side,



LAST DUEL



JOURNEY TO THE CENTER OF THE EARTH



U.S. GOLD



JOURNEY TO THE CENTER OF THE EARTH



You can't always see what you're up against until it's on top of you. There are bottomless pits that have to be avoided, and occasionally you're able to go through a panel that will allow limited travel on the barren landscape that flanks each side (a landscape that flashes on and off to add to the general confusion.) Later, your car will transform into a galactic Hoverplane. For the most part, though, it's business as usual: trying to survive.

Your vehicle isn't too sturdy though. Scrape a wall or receive a hit or two, and it flames and dies. Nor can you recover from the bottomless pits (figuratively or literally); you just get a new life and car. There are blasting gun stations, mobile-seeking missiles and descending bombs. The biggest troublemaker is the stationary bazooka platform (it looks more like three guys melted together wearing paint buckets). When it senses your position and rotates its big, bad barrel, plan on many direct hits before it shuts down. Which is why picking up a rapid-fire icon is sound advice. (Good sound effects here by the way; the bazooka is "pleasantly" shrill.)

With plenty of screens to cover, and random jumbling of icons and enemies, the addiction factor of *Last Duel* is an eight.

Journey combines strategy with arcade action: One of the first sequences requires you to dodge boulders as you direct the scientist up a steep underground cliff. Graphics and sound are digitized and combine to create a realistic mood.

Instead of traveling in a car, *Journey to the Center of the Earth* restores power to your feet—and oh, how they will ache before you're done. Choose one of four scientists to assist in the underground voyage, each man having special characteristics. (Sometimes this means being young and tough, but not so smart; or a tad older and wiser, but clumsy.) *Journey* combines strategy with arcade action: One of the first sequences requires you to dodge boulders as you direct the scientist up a steep underground cliff (expect worse before it's over). Graphics and sound are digitized and combine to create a realistic mood.

The control panel allows a multitude of choices. It is here that your portrait resides (which might get a bit seedy as things progress). A record of time and date is tucked in above a cutaway map of your itinerary (established during a previous

expedition that must be proven out). A series of meters dots the bottom, recording the status of food and water supplies, vitality, physical and general condition. These can all be affected by obvious actions, such as sleeping or searching for water. (No water means no life.)

One of the more interesting functions is Study. This brings up screens that display the formation of the earth's crust at the present location. These screens are informative both for the game and as a source of general knowledge.

Compass arrows on the control panel move you about; digitized footsteps sound and on-screen prompts indicate what's happening. Should you become damaged (polite way of saying hurt), better go to the first-aid screen and do what you can. Other arcade sequences exist as well. There's much to *Journey*—some of it best left unsaid. You like surprises, don't you?

Moving over to the area of productivity gives us *K-Scope* and *K-Spect*, from Kuma Computers. It's for those who enjoy "nuts-'n-bolts" magazines (and taking apart alarm clocks to see how they tick). These two programs combine with a hardware interface that enables the ST to be used as an audio-frequency oscilloscope or a spectrum analyzer, respectively.

Software control over the interface's two ports activates such functions as channel selection and frequency response. Results are easily printed out. Additionally, the data can be saved for later display. We'll take a closer look at this next time. Meanwhile, hang loose, stay cool and keep computing. ■



Marshal M. Rosenthal is a New York-based photographer and writer specializing in children's products, videographic enhancements and high-tech entertainment.

PRODUCT INFORMATION:

- *Journey to the Center of the Earth*
- *Last Duel*
- *Road Blaster*
- *U.S. Gold*
- Units 2 & 3 Halford Way
Halford, Birmingham, England B6 7AX
- *K-Scope/K-Spect*
- Kuma Computers Ltd.
12 Horseshoe Park
Pangbourne, Berks, England

ST-LOG SEPTEMBER 1988

Attention Programmers!

ST-LOG Magazine is interested in programs, articles, and software review submissions dealing with the Atari home computers. If you feel that you can write as well as you can program, then submit those articles and reviews that have been floating around in your head, awaiting publication. This is your opportunity to share your knowledge with the growing family of Atari computer owners.

All submissions for publication, both program listings and text, should be provided in printed and magnetic form. A typed or printed copy of text is mandatory and should be in upper and lower case with double spacing. By submitting articles to **ST-LOG Magazine**, authors acknowledge that such materials, upon acceptance for publication, become the exclusive property of **ST-LOG Magazine**. If not accepted for publication, the articles and/or programs will remain the property of the author. If submissions are to be returned, please supply a self-addressed, stamped envelope. All submissions of any kind must be accompanied by the author's full address and telephone number.

Send your programs and articles to:
ST-LOG Magazine
 P.O. Box 1413-M.O.
 Manchester, CT 06040-1413

C-manship:

By Clayton Walnum

This month, as promised, we'll take a look at the second part of the source code for *MicroCheck ST*. But before we get into that, those of you who don't receive the disk version of *ST-LOG* should turn your attention to Listing 1. In order to run *MicroCheck ST*, you need to have, in the same directory as the main program, the file *MICROCHK.RSC*. As you should know, this file contains the data GEM needs to set up the menu bar and the various dialog boxes the program will access. Listing 1 is an *ST BASIC* program that will create the *MICROCHK.RSC* file for you. Type it in, then check your typing using *ST Check*, found elsewhere in this issue. Once you're sure the program has been typed correctly, load it into *ST BASIC* and run it. *MicroCheck*'s resource file will be written to the disk in Drive A.

Back to the task at hand

In the last installment of *C-manship*, we looked at the portion of the *MicroCheck ST* source code that declared all of the global variables used in the program and initialized both GEM and the program itself. Essentially, you could say that the first part of the source code did all the start-up housekeeping. With all the housekeeping (or most of it) out of the way, we are ready to explore the program itself.

Recall that a GEM application is controlled by "events." Whenever the user attempts any form of input, whether it be accessing the menu bar, typing a key, clicking the mouse or fiddling with one of a window's controls, GEM sends our application a message that describes what the user wants to do. It's up to us as programmers to decide how to handle and interpret these messages. We can even ignore them if we want to.

The source code we looked at last time included a function called `get_event()`, which used a call to the function `event_multi()` to retrieve events from GEM and route them to the appropriate parts of the program. In *MicroCheck ST* we are interested in only three types of events: keyboard events, mouse-button events or GEM messages. These three events are handled in *MicroCheck ST* by the functions `handle_keys()`, `handle_button()` and `handle_messages()`, respectively.

Last time, we represented these functions in our source code as "stubs," doing nothing functions that simply fulfill the linker's need for a label. This allowed us to compile and link a small portion of the program. Without the stubs, the linker would have complained about the missing functions and not allowed us to get a running program. This month we'll be replacing the `handle_messages()` stub with the real function, adding to *MicroCheck* the ability to handle GEM message events.

At this point you should load the portion of *MicroCheck ST* we looked at last time and delete the `handle_messages()`

A Complete GEM Application, Part 2

stub. You should then add the code shown in Listing 2. Once Listing 2 has been merged with the code from last time, you will be able to compile and link it. When you run the resulting program, rather than ending up with a mostly empty screen as we did in the previous program segment, the addition of the `handle_messages()` function and some of its subordinate functions allows the window to be filled in and the buttons at the bottom of the screen to be drawn. We'll see why soon.

Unfortunately, we still haven't added a quit function to the program, so once you have run the program, the only way to get back to the desktop is to turn off your computer.

Function `handle_messages()`

As we've discussed before (see the June '87 *C-manship*), when GEM sends a message, it stores it in an array that we must provide. In *MicroCheck ST*, this array is named `msg_buff[]`. The array element `msg_buff[0]` will contain the type of message we've received.

Now take a look at the top of Listing 2, where you'll find the function `handle_messages()`. This function simply takes the message type that was stored in `msg_buff[0]` and uses it in a `switch` statement to route the message to the appropriate part of the program. The `switch` is followed by seven case statements that handle menu messages, redraw messages, window-filled messages, arrow messages, vertical-slider messages, horizontal-slider messages and window-closed messages, respectively.

This month we will be dealing only with redraw messages and window-filled messages. The functions `do_redraw()` and `do_full()` handle those. The other functions needed for `handle_messages()`—`do_menu()`, `do_arrow()`, `do_vslide()`, `do_hslide()` and `do_window_close()`—are represented in this month's code segment

by stubs.

Notice that in the call to `do_redraw()`, we are passing the address of `msg_buff[4]` cast into a pointer to `GRECT` (a structure that is defined in the header file `OBDEFS.H` and that contains the coordinates for a GEM rectangle). The array elements `msg_buff[4]` through `msg_buff[7]` contain the coordinates for the work area of the window that needs to be redrawn. (See the September '87 *C-manship*.)

Function `do_redraw()`

This function steps through the rectangle list, redrawing all portions of the windows that need to be refreshed. This includes the "invisible" window 1 mentioned in the last *C-manship*. The information buttons on the bottom of the screen are drawn as a result of a redraw message for this window.

The function `do_redraw()`, after locking the windows (to prevent corruption by such occurrences as a menu bar dropping down), steps through the rectangle list for each window, calculating the rectangles that need to be redrawn. (See the September '87 and April '88 *C-manships*.) The redraw itself is handled by the function `draw_interior()`, which we'll get to in a moment. After the rectangle list is empty, we unlock the window and control returns back to `get_event()`, where we wait for another event.

Function `draw_interior()`

This function handles the actual redraw operations. The structure `clip`, passed to this function from `do_redraw()`, contains the coordinates of the rectangle to be redrawn. First we turn off the mouse and set the drawing mode to "replace." Then, if the redraw message is for window #1, the invisible window, we calculate the coordinates, based on the current resolution, of the portion of the window to be redrawn, and redraw both that rectangle and the boxes on the bottom of the

screen. (The only rectangle that'll ever be redrawn in the invisible window is the section of the screen below the visible window, the part of the screen that contains the buttons. If the visible window, the one that displays the check data, is fully opened, we'll never get a redraw message for the invisible window since it is completely covered.)

If the redraw message isn't for window #1, it has to be for window #2, the visible window. First we check the status of the flag `full_draw`. If its value is `FALSE`, we are not going to be updating the entire window, so we set a clipping rectangle to the portion of the window's work area we are going to redraw. A call to `draw_rect()` redraws the window rectangle.

The check data for *MicroCheck ST* is stored in an array of structures called `checks[]`. The first check is in element 0 of the array, the second in element 1, and so on. The position within the array at which a check is located is its index.

We find out how many lines of check data will fit in the window (`lines_avail`) based on the flag `full`, which indicates whether the window has been fully opened. Next we find out how many lines we need to display (`lines_shown`) by subtracting from the total number of checks (`cur_count`) the index of the check data shown at the top of the window (`cur_top`). (This doesn't have to be the first record in the `checks[]` array; if the user has used the vertical scroll bar or down arrow, the index for the check data at the top of the window could be almost anything, up to the maximum minus the number of checks that'll fit in the window.)

If `lines_avail` is larger than `lines_shown`, we have more space in the window than we have checks to fill it (when starting with `cur_top`), so we calculate a new `cur_top` in an attempt to show as many checks as possible without any blank space at the bottom. If `cur_top` ends up less than 0, we don't have enough check data in the array to fill the window. In this

case, we just set `cur_top` to the first check in the array.

We then set the vertical and horizontal sliders, call `update_chk_wnd()` to print the check data that should appear in the window, turn off the `full_draw` flag, turn off clipping (otherwise we won't be able to print anything to the screen unless its coordinates are within the clipping rectangle) and turn the mouse back on.

Function draw_rec()

This function does nothing more than draw a filled rectangle at the coordinates passed in the structure `rec`. The interior style, fill style and color of the rectangle are also passed to the function, although in most cases when you're redrawing a window, you'll want a solid white rectangle. Passing the extra information makes the function a little more flexible. Notice also that, as usual, whenever we draw anything on the screen, we shut off the mouse. This is important if you want to keep your screen clean. (See the January '87 *Cmanship*.)

Function set_clip()

Here we set the clipping rectangle, the portion of the screen in which we want to restrict our drawing, to the coordinates passed in the structure `rec`. The integer `flag` is a Boolean variable that tells the function whether we're turning the clipping on or off. (TRUE turns it on.)

Function update_chk_wnd()

This function simply prints the necessary check data to the newly updated window (with a call to `prnt_chk_wnd()`). We get the coordinates of the window's work area and use those values to calculate, based on the character size and the check data index, the position within the window in which to print the data. We need to use the character size because high-resolution characters are a different height (16 bits) from medium-resolution characters (eight bits). Remember that the integer `cur_top` holds the index for the check data that should be displayed at the top of the window.

Function prnt_chk_wnd()

It's here that we actually print the check data to the screen. The index into the check array and the row at which the data should be printed are passed to this function from `update_chk_wnd()`. This function uses basic C text handling, so you

should be able to understand it easily. Of special note, however, is the structure `cur_chk_strc` and the use of the flag `left`.

Because *MicroCheck* uses two different structures for holding check data—one for the regular monthly file and one for data obtained with the search function—we use the pointer `cur_chk_strc` to point to the currently active structure. This simplifies handling the two structures, thereby eliminating our need to know whether the program is in the edit or search mode.

The flag `left` indicates the position of the window's horizontal slider. If the slider is all the way to the left (`left` is TRUE), we display all the check data except the date, which is in the right-hand unseen portion of the window. If the slider is to the right, we display the date, but not the check's cancel indicator or number, which are now in the left-hand unseen portion of the window.

Function format_date()

Because the date for a check is stored in the format `mmddyy`, it must be converted to the more typical `mmddyy` format before it is printed in the check window. The function `format_date()` takes care of this chore for us. The pointer `d1` is the address of a character array where the final formatted date will be stored, and the pointer `d2` is the address of the date in its unformatted form.

Function draw_buttons()

Across the bottom of the *MicroCheck* screen there is a series of six boxes that contain various information about the currently opened account. The function `draw_buttons()`, along with a lot of help from its subordinate functions, prints the information in the boxes.

Function set_buttons()

Most of the data to be displayed in the boxes is in integer or long-integer format, so we must first, using `sprintf()`, convert it to strings. Note that the strings to be displayed in the boxes are stored in character arrays whose names end with `_but`.

The raw data to be converted is stored in four global variables: The long integer `balance` is the account's balance, and the integers `num_trans`, `num_chks` and `num_depos` are the total number of transactions, the number of checks and the number of deposits in the currently opened month, respectively.

Function button()

This function does the actual drawing of the boxes. The pointer `str1` is the ad-

dress of the box's label, the pointer `str2` is the address of the information to print, and `x1` is the X coordinate of the box being updated. The function first calculates the position in which the boxes should be drawn. It then, with a little help from `center_buttoning()`, draws the boxes. (If the VDI functions used in this piece of code are confusing, you should review the *Cmanship* in Issue 9 of STLOG, which was bound into the December '86 issue of ANALOG Computing.)

Function button()

This function centers and prints the text in the boxes. A pointer to the text to be printed, along with the X and Y coordinates for the box, is passed to the function from `button()`.

Function do_full()

The check-display window in *MicroCheck* can be opened to only two sizes. The smaller size holds 16 checks and allows the user to view the information boxes on the bottom of the screen. The full-size window can hold up to 20 checks but covers the information boxes.

We keep track of the window's current size with the flag `full`. Whenever we get a window-filled message from GEM (this happens when the user clicks on the full box in the upper right-hand corner of the window), we check the `full` flag and set the window's size accordingly. We are actually concerned only with the window's height, and because a high-resolution screen is twice the vertical resolution of a medium-resolution screen, we use the integer variable `res`, which holds the current screen resolution, to calculate the window's height.

Time for another break

Next time, yet more of *MicroCheck ST*. (I bet you can hardly wait.) ■



Clayton Walnum is the executive editor of STLOG and ANALOG Computing, as well as the associate editor of Videogames & Computer Entertainment.

VideoGames & Computer Entertainment™



12

**ALL COLOR ISSUES
ONLY \$19.95**

Save over \$15 off the cover price!



- GAME REVIEWS
- ARCADE ACTION
- STRATEGY GUIDES
- TECHNICAL REPORTS
- COMPUTER SOFTWARE

VideoGames & Computer Entertainment

P.O. BOX 16927, N. HOLLYWOOD, CA 91615

☐ **Yes!**

Sign me up for 12 issues for only \$19.95—I'll save over \$15!

☐ Payment Enclosed — ☐ Charge My ☐ VISA ☐ MC

#

EXP

NAME

ADDRESS

SIGNATURE

CITY

STATE

ZIP

Money back on unused subscriptions if not satisfied!
Foreign—add \$10 for postage.

Your first issue will arrive in 6 to 8 weeks.
WATCH FOR IT!!

Expires November 29, 1989

CJVWA

BY RICHARD LEINECKER

THE *ULTI*

*a*s I poise the mouse to run *ST Writer*, I hear a voice from the kitchen. "Hey, can you help with the groceries?" Eager to dispel my current computer-addict image, I proceed to the car and bring in the groceries. One thing (submarine sandwich) leads to another (Miller Lite), and it is eight hours later that I return to my ST.



*i*scream. A dim trash can has appeared in the upper right-hand corner of the *ST Writer* menu screen as well as the edit screen. I know that some of my prose is lacking, but what is *ST Writer* trying to tell me? Just this: "Your monitor now has a trash can permanently burned into it because you left it on for eight hours."

Two precautionary measures would have avoided this disaster. The most obvious, of course, would have been to turn the computer off. The other alternative would have been to install a screen-saver program so that if there was no keyboard or mouse input for a period of time, it would take evasive action, shutting off the screen.

MATE SCREEN SAVER

UTILITY
ANY RESOLUTION

the next day I installed a screen-saver program, and it worked like a charm. When I did not hit a key or move the mouse for about ten minutes, the screen went black. Sometimes, however, I found myself reaching over to turn on a computer that was simply blacked out by the screen-saver program.

Then, in a computer shop, I saw a Mac II that had been idle for a while. It seemed to be showing a fireworks demo. The patterns were randomly generated, creating an interesting and varied series of explosions. I became addicted to the show as I awaited the next unique pattern—until I hit a key and instantly saw the Mac II desktop. The salesman informed me that what I had seen was simply a screen-saver program.

During the next six months, I worked on a commercial program that loaded in a lot of compressed data. Anyone who has loaded a compressed picture in *DEGAS Elite* knows how slow this process can be. I needed to find a method of entertaining the user during the loading, and the idea of using a fireworks effect struck me. I implemented this idea, and the beta-testers found it interesting. No more boredom while data loaded. I knew then what I had to do: develop a screen saver like the one I had seen on the Mac.

Typing it in

Listing 1 is an ST BASIC program that'll create your copy of the *Ultimate Screen Saver*. Type it in, then use *ST Check* (found elsewhere in this issue) to check your work. When you're sure the program is typed properly, run it. The *SAVER.PRG* program file will be written to the disk in Drive A.

Listing 2 is the assembly language source code for the *Ultimate Screen Saver*. You don't need to type it in. It is included only for those people who are interested in assembly language programming.

Using the program

The program *SAVER.PRG* can either be put in your auto folder or run from desktop. The actual code stays resident in the computer and is installed as a vertical blank interrupt. If there are no more interrupt vector slots available, the program will not install. The fireworks will be invoked after about a nine-minute period, after which there is no keyboard event or mouse movement. Any key press or mouse movement resets the counter or resets the original screen.

I hope you enjoy the show! ■



Richard Leinecker is a math instructor and the senior programmer for IntraCorp, Inc., in Miami, Florida. He has published a number of software titles, including a gambling simulation, and several books containing hardware projects for the ST.

SCREEN SAVER

Listing 1:
ST BASIC

THE ULTIMATE SCREEN SAVER

```

100 OPEN "R", #1, "A:SAVER.PRGM", 16:FIELD#
1, 16 AS BS
110 AS="":FOR I=1 TO 16:READ US:IF US=
"X" THEN 140
120 A=VAL("&H"+US):PRINT "X":AS=AS+CH
R$(A):NEXT
130 LSET BS=AS:R=R+1:PUT 1,R:GOTO 110
140 CLOSE 1:PRINT:PRINT "ALL DONE!"
1000 data 60,1A,00,00,07,78,00,00,00,0
0,00,00,84,72,00,00
1010 data 04,52,00,00,00,00,00,00,00,0
0,00,00,24,4F,2E,7C
1020 data 00,00,00,8C,96,2F,3C,00,00,00,7
E,3F,3C,00,26,4E,4E
1030 data 5C,8F,3F,3C,00,22,4E,4E,54,8
F,0D,8C,00,00,20
1040 data 20,40,23,00,00,00,08,30,20,7
C,00,00,06,68,01,FC
1050 data 00,00,00,00,02,20,89,00,00,08,3
0,20,40,20,8C,00,00
1060 data 05,E4,20,3C,00,00,08,5E,02,8
0,FF,FF,FE,00,06,80
1070 data 00,00,02,00,23,C0,00,00,88,7
E,2A,60,00,04,20,20
1080 data 00,0C,00,AD,00,14,D0,AD,00,1
C,D0,8C,00,00,01,00
1090 data 42,67,2F,00,3F,3C,00,31,4E,4
1,32,3C,00,00,20,7C
1100 data 00,00,04,56,22,50,30,3C,00,0
0,33,C0,00,00,88,85
1110 data 24,7C,00,00,04,54,36,12,D3,F
C,00,00,00,04,24,11
1120 data 0C,82,00,00,00,00,67,00,00,1
2,52,41,82,43,67,00
1130 data 00,12,D3,FC,00,00,00,04,60,E
4,25,8C,00,00,00,63
1140 data 4E,75,4E,75,30,39,00,00,88,8
6,8C,40,00,01,66,00
1150 data 02,76,20,7C,00,00,07,D8,22,7
C,00,00,88,88,4E,B9
1160 data 00,00,07,68,4E,B9,00,00,01,8
0,22,7C,00,00,07,D8
1170 data 20,7C,00,00,88,88,4E,B9,00,0
0,07,68,20,7C,00,00
1180 data 07,EC,22,7C,00,00,88,88,4E,B
9,00,00,07,68,4E,B9
1190 data 00,00,01,80,22,7C,00,00,07,E
C,20,7C,00,00,88,88
1200 data 4E,B9,00,00,07,68,20,7C,00,0
0,08,00,22,7C,00,00
1210 data 88,88,4E,B9,00,00,07,68,4E,B
9,00,00,01,80,22,7C
1220 data 00,00,00,00,20,7C,00,00,88,8
8,4E,B9,00,00,07,68
1230 data 20,7C,00,00,00,14,22,7C,00,0
0,88,88,4E,B9,00,00
1240 data 07,68,4E,B9,00,00,01,80,22,7
C,00,00,00,14,20,7C
1250 data 00,00,88,88,4E,B9,00,00,07,6
8,4E,75,30,39,00,00
1260 data 88,94,0C,40,00,00,66,00,00,7
C,30,39,00,00,88,88
1270 data 32,39,00,00,88,90,34,39,00,0
0,08,92,4E,B9,00,00
1280 data 06,6E,32,39,00,00,88,90,34,3
9,00,00,88,92,53,41
1290 data 38,39,00,00,88,8C,0C,44,00,0
0,66,00,00,88,52,42
1300 data 60,00,00,0C,0C,44,00,03,66,0
0,00,04,53,42,33,62
1310 data 00,00,88,92,33,C1,00,00,88,9
0,82,79,00,00,88,8E
1320 data 6E,00,01,56,33,FC,00,00,00,0
0,88,96,33,FC,00,01
1330 data 00,00,80,94,33,FC,00,00,00,0
0,88,98,4E,B9,00,00
1340 data 07,0A,4E,75,30,39,00,00,88,9
8,52,40,0C,40,00,04
1350 data 67,00,00,0A,33,C0,00,00,88,9
8,4E,75,33,FC,00,00
1360 data 00,00,88,98,4E,B9,00,00,07,0
A,30,39,00,00,88,94
1370 data 52,40,33,C0,00,00,88,94,0C,4
0,00,09,67,00,00,0A
1380 data 4E,B9,00,00,07,0A,4E,75,30,3

```

```

9,00,00,88,96,52,40
1390 data 33,C0,00,00,88,96,32,39,00,0
0,88,8A,82,40,67,00
1400 data 00,64,33,FC,00,01,00,00,88,9
4,20,7C,00,00,04,8C
1410 data 30,10,32,39,00,00,08,56,C0,4
1,0C,40,00,00,66,00
1420 data 00,06,00,40,00,01,33,C0,00,0
0,88,88,30,10,02,40
1430 data 00,0F,30,10,02,43,00,07,32,3
9,00,00,88,92,34,39
1440 data 00,00,88,90,04,41,00,07,04,4
2,00,04,D2,40,D4,43
1450 data 33,C1,00,00,88,92,33,C2,00,0
0,88,90,4E,B9,00,00
1460 data 07,00,4E,75,33,FC,00,00,00,0
0,88,94,20,7C,00,00
1470 data 04,8C,30,10,32,39,00,00,08,5
6,C0,41,0C,40,00,00
1480 data 66,00,00,06,00,40,00,01,33,C
0,00,00,88,88,38,10
1490 data 02,44,00,1F,06,44,00,20,33,C
4,00,00,88,8E,38,10
1500 data 02,44,00,03,52,44,33,C4,00,0
0,88,8A,38,10,02,44
1510 data 00,03,33,5A,00,00,88,8C,C8,F
9,00,00,88,58,08,7F
1520 data 00,00,00,54,33,FC,00,C7,00,0
0,88,90,32,3C,00,C7
1530 data 33,C4,00,00,88,92,34,04,30,3
9,00,00,88,88,4E,B9
1540 data 00,00,06,6E,4E,75,30,39,00,0
0,88,82,52,40,33,FC
1550 data 00,00,88,82,0C,40,70,00,66,0
0,02,84,20,39,00,00
1560 data 88,7E,20,40,30,3C,00,00,20,F
C,00,00,00,52,40
1570 data 0C,40,1F,40,66,F2,20,7C,00,0
0,04,4E,23,00,00
1580 data 0C,2C,20,7C,FF,82,01,22,7
C,00,00,88,7E,D3,FC
1590 data 00,00,00,01,10,91,D1,FC,00,0
0,00,02,D3,FC,00,00
1600 data 00,00,01,10,91,20,7C,00,FF,82,6
0,10,10,02,40,03
1610 data 33,C0,00,00,08,28,33,FC,00,0
0,00,00,88,2A,0C,40
1620 data 00,01,66,00,00,2A,20,7C,00,F
F,82,60,32,10,33,C1
1630 data 00,00,00,2A,30,3C,00,00,20,7
C,00,00,04,4C,10,80
1640 data 20,7C,00,FF,82,60,10,80,33,C
0,00,00,88,28,0C,40
1650 data 00,00,66,00,00,2E,33,FC,00,0
8,00,00,88,54,33,FC
1660 data 00,00,00,00,00,88,56,33,FC,00,0
0,00,00,88,53,33,FC
1670 data 00,28,00,00,08,5A,33,FC,00,0
1,00,00,88,5C,60,00
1680 data 00,2A,33,FC,00,02,00,08,5
4,33,FC,00,00,00
1690 data 00,00,33,FC,00,0A,00,00,08,5
8,33,FC,00,28,00,00
1700 data 08,5A,33,FC,00,02,00,08,5
C,22,7C,00,00,08,34
1710 data 20,7C,00,FF,82,40,4E,B9,00,0
0,06,FA,20,7C,00,00
1720 data 07,98,22,7C,00,FF,82,40,4E,B
9,00,00,06,FA,33,FC
1730 data 00,01,00,00,88,86,33,FC,00,0
0,00,00,88,84,3E,3C
1740 data 00,03,33,FC,00,00,00,88,9
4,33,FC,00,00,00,00
1750 data 88,96,20,7C,00,00,04,8C,30,1
0,32,39,00,00,08,56
1760 data C0,41,0C,40,00,00,66,00,00,0
6,00,40,00,01,33,C0
1770 data 00,00,88,88,88,30,72,44,C0,F
4,00,00,06,4E,00,00,00
1780 data 33,C4,00,00,88,8E,38,10,02,4
4,00,03,52,44,33,C4
1790 data 00,00,88,88,38,07,33,C4,00,0
0,88,8C,C8,F9,00,00
1800 data 00,00,88,79,00,00,88,5A,33,FC
0,C0,C7,00,00,88,90

```

1810 data 32,3C,00,C7,33,C4,00,00,88,9
 2,34,04,3E,39,00,00
 1820 data 88,84,52,47,33,C7,00,00,88,8
 4,0C,47,00,01,5E,00
 1830 data 00,30,20,7C,00,00,88,88,22,7
 C,00,00,07,D8,4E,B9
 1840 data 00,00,07,68,30,39,00,00,88,8
 8,32,39,00,00,88,90
 1850 data 34,39,00,00,88,92,4E,B9,00,0
 0,06,6E,60,00,FF,54
 1860 data 0C,47,00,02,6E,00,00,30,20,7
 C,00,00,88,88,22,7C
 1870 data 00,00,07,EC,4E,B9,00,00,07,6
 8,30,39,00,00,88,88
 1880 data 32,39,00,00,88,90,34,39,00,0
 0,88,92,4E,B9,00,00
 1890 data 06,6E,60,00,FF,1E,0C,47,00,0
 3,6E,00,00,30,20,7C
 1900 data 00,00,88,88,22,7C,00,00,00,0
 0,4E,89,00,00,07,68
 1910 data 30,39,00,00,88,88,32,39,00,0
 0,88,90,34,39,00,00
 1920 data 88,92,4E,B9,00,00,06,6E,60,0
 0,FE,E8,0C,47,00,04
 1930 data 66,00,00,2C,20,7C,00,00,88,8
 8,22,7C,00,00,08,14
 1940 data 4E,39,00,00,07,68,30,39,00,0
 0,88,88,32,39,00,00
 1950 data 88,90,34,39,00,00,88,92,4E,B
 9,00,00,06,6E,4E,75
 1960 data 33,FC,00,00,00,88,82,48,E
 7,01,C0,30,39,00,00
 1970 data 00,30,00,40,00,01,66,00,00,6
 8,20,7C,FF,8E,82,01
 1980 data 22,7C,00,00,08,2C,D3,FC,00,0
 0,00,01,10,91,D1,FC
 1990 data 00,00,00,02,D3,FC,00,00,00,0
 1,10,91,33,FC,00,00
 2000 data 06,FC,00,94,22,7C,00,FF,82,4
 0,20,7C,00,00,08,34
 2010 data 4E,89,00,00,06,FA,33,FC,00,0
 0,00,00,88,86,30,39
 2020 data 00,00,08,2A,0C,40,00,00,67,0
 0,00,16,30,FC,00,01
 2030 data 20,7C,00,00,04,4C,10,80,20,7
 C,00,FF,82,60,10,80
 2040 data 4C,0F,03,81,4E,F9,11,11,11,1
 1,26,79,00,00,88,7E
 2050 data C2,FC,00,A0,02,81,00,00,FF,F
 F,07,C1,02,82,00,00
 2060 data FF,FF,8A,FC,00,10,22,02,02,8
 2,00,00,FF,FF,C4,F9
 2070 data 00,00,00,54,02,82,00,00,FF,F
 F,07,C2,00,89,EE,89
 2080 data 82,81,00,00,FE,28,7C,00,0
 0,07,70,00,07,3E,00
 2090 data 08,00,00,00,67,00,00,08,34,1
 3,87,42,36,82,D6,FC
 2100 data 00,02,08,00,00,01,67,00,00,0
 8,34,13,87,42,36,82
 2110 data 06,FC,00,02,08,00,00,02,67,0
 0,00,08,34,13,87,42
 2120 data 36,82,08,00,00,03,67,00,00,0
 C,D6,FC,00,02,34,13
 2130 data 87,42,36,82,4E,75,3E,3C,00,0
 0,32,D8,52,47,0C,47
 2140 data 0C,47,00,06,FE,4E,75,30,39,00,0
 0,88,88,3E,3C,00,00
 2150 data 24,7C,00,00,07,88,3C,39,00,0
 0,88,94,3A,06,0C,FC
 2160 data 00,03,34,39,00,00,88,92,94,4
 6,38,39,00,00,88,5C
 2170 data 0C,47,00,02,66,00,00,04,94,4
 6,32,39,00,00,88,90
 2180 data 92,4E,38,1D,C8,C5,C8,F9,00,0
 0,08,5C,D4,44,38,1D
 2190 data C8,C5,D2,44,52,47,4E,89,00,0
 0,06,6E,00,07,68,30
 2200 data 66,00,4E,75,30,3C,00,00,32,0
 8,52,40,0C,40,00,0A
 2210 data 66,FE,4E,75,80,00,40,00,20,0
 0,10,00,88,00,04,00
 2220 data C8,C0,01,00,00,00,40,00,2
 0,00,10,00,08,86,04
 2230 data 00,02,00,01,00,00,00,70,07,0

0,07,00,07,70,07,07
 2240 data 00,77,07,77,07,70,07,04,00,7
 4,07,40,07,04,07,40
 2250 data 04,00,07,00,00,00,03,00,0
 1,00,01,00,03,00,00
 2260 data 00,05,00,01,00,06,00,03,00,0
 5,00,05,00,03,00,06
 2270 data 00,01,00,05,00,00,00,00,00,0
 0,00,00,00,00,00,00
 2280 data 00,00,00,00,00,00,00,00,00,0
 0,00,00,00,00,00,00
 2290 data 00,00,00,00,00,00,00,00,00,0
 0,00,00,00,00,00,00
 2300 data 00,00,00,00,00,00,00,00,00,0
 0,00,00,00,00,00,00
 2310 data 00,00,00,00,00,00,00,00,00,0
 0,00,00,00,00,00,00
 2320 data 00,00,00,00,48,62,64,76,62,6
 1,73,65,C0,00,00,00
 2330 data 00,22,53,75,70,65,78,65,63,0
 0,C0,00,00,00,26
 2340 data 50,74,65,72,60,72,65,73,C0,0
 0,00,00,00,31,47,45
 2350 data 40,44,4F,53,00,00,C0,00,00,0
 0,00,01,58,42,49,4F
 2360 data 53,00,00,00,C0,00,00,00,00,0
 E,73,74,61,72,74,00
 2370 data 00,00,82,00,00,00,00,00,75,7
 3,74,68,00,00,00,00
 2380 data 81,00,00,0C,96,73,65,74,7
 5,70,00,00,00,82,00
 2390 data 00,00,00,7E,68,65,79,76,65,6
 3,74,6F,81,00,00,00
 2400 data 00,30,69,6E,73,65,72,74,00,0
 0,82,00,00,00,06,68
 2410 data 74,72,61,70,70,65,72,00,82,0
 0,00,00,05,E4,6E,65
 2420 data 77,60,65,60,00,00,81,00,00,0
 0,08,5E,6E,65,77,73
 2430 data 63,74,65,65,81,00,00,00,88,7
 E,76,62,69,6E,6E,61
 2440 data 67,00,81,00,00,08,86,63,6
 8,65,63,68,00,00,00
 2450 data 82,00,00,00,00,A2,69,6E,73,7
 4,61,6C,6C,00,82,00
 2460 data 00,00,00,0E,73,68,69,70,00,0
 0,00,00,82,00,00,00
 2470 data 00,C6,76,62,69,00,00,00,00,0
 0,82,00,00,00,00,C8
 2480 data 63,6F,6E,74,69,6E,75,65,82,0
 0,00,00,83,4A,66,77
 2490 data 31,00,00,00,00,00,84,00,00,0
 0,07,08,66,63,6F,6C
 2500 data 6F,72,00,00,81,00,00,00,88,8
 8,61,73,73,69,67,6E
 2510 data 00,00,82,00,00,00,07,68,72,6
 5,64,6C,76,62,69,00
 2520 data 82,00,00,00,01,80,66,77,32,0
 0,00,00,00,00,84,00
 2530 data 00,00,07,EC,66,77,33,00,00,0
 0,00,00,84,00,00,00
 2540 data 08,00,66,77,34,00,00,00,00,0
 0,84,00,00,00,08,1
 2550 data 66,6E,75,6D,00,00,00,00,81,0
 0,00,00,88,94,6D,61
 2560 data 68,65,66,77,00,00,82,00,00,0
 0,02,08,74,79,00,00
 2570 data 00,00,00,81,00,00,00,88,9
 0,74,78,00,00,00,00
 2580 data 00,00,81,00,00,00,88,92,70,6
 9,70,65,6C,00,00,00
 2590 data 82,00,00,00,06,6E,74,72,61,6
 A,65,63,74,6F,81,00
 2600 data 00,00,88,8C,6E,6F,74,6C,65,6
 6,74,00,82,00,00,00
 2610 data 01,C8,6E,6F,74,73,69,64,65,0
 0,82,00,00,01,02
 2620 data 66,68,65,69,67,68,74,00,81,0
 0,00,00,88,8E,6D,61
 2630 data 68,65,70,69,78,65,82,00,00,0
 0,03,3C,6E,75,6D,63
 2640 data 6F,75,6E,74,81,00,00,00,88,9
 6,66,77,76,62,6C,61
 2650 data 6E,68,81,00,00,00,88,98,64,7
 2,61,77,66,77,00,00

```

4,12,0E,08,06,06,08
3090 data 0A,08,08,0E,06,06,06,06,06,0
6,12,06,06,06,06,06
3100 data 06,12,06,06,06,06,06,12,0
6,06,06,06,06,06,0A
3110 data 0A,14,1E,0C,06,08,06,2C,24,1
8,60,0A,06,C,08,10
3120 data 0C,10,00,00,00,00,00,00,00,0
0,00,00,00,00,00,00
3130 data *

```

100 data 572, 544, 391, 421, 536, 632, 599, 7
 86, 873, 785, 6119
 1050 data 643, 928, 699, 905, 729, 672, 811
 595, 847, 723, 7521
 1150 data 819, 339, 748, 890, 737, 690, 822,
 750 data 1610
 1250 data 743, 714, 742, 723, 676, 660, 816,
 765, 788, 707, 7362
 1350 data 801, 745, 712, 770, 800, 771, 664,
 645, 633, 727, 7285
 1450 data 827, 700, 720, 669, 723, 706, 898,
 813, 809, 759, 7695
 1550 data 681, 723, 787, 28, 805, 704, 722, 7,
 83, 650, 802, 6685
 1650 data 756, 766, 690, 725, 799, 724, 878,
 747, 755, 735, 7880
 1750 data 693, 693, 805, 779, 868, 866, 780,
 747, 818, 693, 7813
 1850 data 840, 714, 793, 761, 781, 729, 691,
 802, 705, 733, 7649
 1950 data 836, 840, 880, 895, 768, 759, 827,
 750 data 827
 2050 data 11, 103, 32, 815, 791, 637, 749, 72,
 8, 847, 769, 5482
 2150 data 897, 685, 667, 982, 895, 794, 674,
 523, 544, 641, 7302
 2250 data 827, 700, 744, 469, 470, 464, 645,
 708, 705, 765, 5596
 2350 data 719, 682, 608, 718, 765, 703, 783,
 779, 813, 710, 7280
 2450 data 785, 682, 702, 825, 711, 780, 70,
 750 data 896, 896, 700, 700, 700, 700, 700,
 2550 data 747, 678, 615, 701, 786, 787, 829,
 810, 816, 852, 7621
 2650 data 789, 790, 730, 865, 848, 834, 770,
 813, 794, 747, 7980
 2750 data 827, 700, 761, 714, 722, 811, 732,
 755, 796, 773, 7741
 2850 data 546, 806, 799, 714, 726, 743, 791,
 776, 800, 803, 7843
 2950 data 805, 890, 861, 862, 814, 791, 601,
 592, 556, 613, 7305
 3050 data 616, 646, 626, 640, 591, 565, 665,
 484, 197, 5029

**FILL
OUT COUPON ON
PAGE 35...!**

```
; THE ULTIMATE SCREEN SAVER
; by Richard Leinecker
; Copyright 1989 by ST-LOG
```

SCREEN SAVER Listing 2: Assembly

```
Kbdbufase    = $22
Supexec      = $26
Ptermres     = $31

GEMDOS       = 1
XBIO$        = 14
```

```
*****
***** start of program *****
*****
```

```
.text

start:
    movea.l a7,a5          ; copy a7 for base page calc
    movea.l #ustk,a7       ; init adr for local stack

    move.l #setup,-(sp)    ; put address of set up routine on stack
    move.w #Supexec,-(sp)  ; SUPER mode command
    trap    #XBIO$        ; execute setup routine in SUPERVISOR mode
    addq.l  #6,sp          ; restore the stack pointer
```

```
*****
*          Insert keyboard patch
*****
```

```
    move.w #Kbdbufase,-(sp) ; XBIO$ to get vectors
    trap    #XBIO$         ; do the trap
    addq.l  #2,sp           ; clean up stack
    add.l   #32,d0          ; +32 to get the keyboard vector address
    movea.l d0,a0          ; put it in a0
    movea.l {a0},keyvector ; move the vector to keyvector

    movea.l #insert,a0     ; address of the jmp in the patch
    addq.l  #2,a0          ; adjust it
    movea.l keyvector,{a0} ; now put the keyboard vector in place

    movea.l d0,a0          ; get address of patch
    movea.l #trapper,{a0}  ; install it

    movea.l #newmem,d0     ; new screen memory into d0
    andi.l  #$ffffe00,d0   ; assure 512 byte boundary
    addi.l  #512,d0        ;
    movea.l d0,newscreen   ; put this value in newscreen
```

```
*****
*          calculate amount of memory to keep, then terminate!
*****
```

```
    movea.l 4(a5),a5       ; basepage address
    movea.l $(a5),d0       ; length of text segment
    add.l   $14(a5),d0     ; length of data segment
    add.l   $1c(a5),d0     ; length of bss segment
    add.l   #$100,d0       ; allow for basepage

    clr.w   -(sp)          ; specify exit code
    movea.l d0,-(sp)       ; length of block to keep
    move.w  #Ptermres,-(sp) ; Terminate and stay resident
    trap    #GEMDOS        ; This doesn't return!
```

```
*****
*          Install the vertical blank interrupt
*****
```

```
setup:
    move.w #0,d1           ; set the counter to zero
    movea.l #$456,a0       ; load a0 with address of the pointer to the vector table
    movea.l {a0},a1       ; load a1 with the address of the vector table

    move.w #0,d0           ; zero the color
    move.w d0,vbiflag      ; set the flag to 0

    movea.l #$454,a2       ; nvbls
    move.w  {a2},d3        ; get the number of vector slots

    adda.l  #4,a1

check:
    movea.l {a1},d2        ; put the vector address in d2
    cmpi.l #0,d2          ; see if it is a null
```



```

beq      install      ; if it is null then go to the install phase
addq.w   #1,d1         ; add one to the counter
cmp.w    d3,d1         ; see if all 8 slots have been checked
beq      skip         ; skip it if there are no slots left
adda.l   #4,a1         ; increment a1 to the next vector slot
bra      check        ; go back and check the next vector slot

install:
move.l   #vbi,(a1)    ; install the vbi address in the vacant slot
rts      ; then leave

skip:
rts      ; leave

*****
*                      Here is the interrupt code                      *
*****

vbi:
move.w   vbiFlag,d0    ; see if the fireworks are going off
cmpl.w   #1,d0         ; 1 means yes
bne      continue     ; if not, keep counting

; firework 1
movea.l  #fw1,a0        ; copy firework data to the
movea.l  #fcolor,a1     ; work area - first assign the
jsr      assign         ; pointers then jsr 'assign'
jsr      realvbi        ; do the actual firework stuff
movea.l  #fw1,a1        ; copy the work area back to
movea.l  #fcolor,a0     ; the firework data
jsr      assign

; firework 2
movea.l  #fw2,a0        ; copy firework data to the
movea.l  #fcolor,a1     ; work area - first assign the
jsr      assign         ; pointers then jsr 'assign'
jsr      realvbi        ; do the actual firework stuff
movea.l  #fw2,a1        ; copy the work area back to
movea.l  #fcolor,a0     ; the firework data
jsr      assign

; firework 3
movea.l  #fw3,a0        ; copy firework data to the
movea.l  #fcolor,a1     ; work area - first assign the
jsr      assign         ; pointers then jsr 'assign'
jsr      realvbi        ; do the actual firework stuff
movea.l  #fw3,a1        ; copy the work area back to
movea.l  #fcolor,a0     ; the firework data
jsr      assign

; firework 4
movea.l  #fw4,a0        ; copy firework data to the
movea.l  #fcolor,a1     ; work area - first assign the
jsr      assign         ; pointers then jsr 'assign'
jsr      realvbi        ; do the actual firework stuff
movea.l  #fw4,a1        ; copy the work area back to
movea.l  #fcolor,a0     ; the firework data
jsr      assign

rts      ; leave, all 4 fireworks done

realvbi:
move.w   fnum,d0        ; see if there are fireworks going off
cmpl.w   #0,d0          ; are there any?
bne      makefw         ; if so, go there

move.w   fcolor,d0      ; load the color
move.w   ty,d1          ; load the y coordinate
move.w   tx,d2          ; load the x coordinate
jsr      pixel         ; XOR the old pixel

move.w   ty,d1          ; load the y coordinate
move.w   tx,d2          ; load the x coordinate
subq.w   #1,d1          ; decrement by delta y
move.w   trajectory,d4  ; load the trajectory number
cmpl.w   #0,d4          ; is it zero?
bne      notleft        ; is it one?
addq.w   #1,d2
bra      notside

```

```

notleft:
    cmpi.w    #3,d4
    bne       notside
    subq.w    #1,d2

notside:
    move.w    d2,tx      ; save tx
    move.w    d1,ty      ; save ty

    cmp.w     fheight,d1 ; compare with the intended height
    bgt       makepixel  ; if not there, just make a pixel

    move.w    #0,numcount ; zero the explosion counter
    move.w    #1,fnum      ; set fnum to the small pattern
    move.w    #0,fwvblanks ; zero the counter
    jsr       drawfw       ; jump and XOR the fw explosion
    rts          ; leave

makefw:
    move.w    fwvblanks,d0 ; load the counter
    addq.w    #1,d0         ; increment it
    cmpi.w    #4,d0         ; see if a delay has passed
    beq       delaygone    ; go on if so
    move.w    d0,fwvblanks  ; store the value
    rts          ; leave

delaygone:
    move.w    #0,fwvblanks ; zero the counter
    jsr       drawfw       ; jump and XOR the fw explosion
    move.w    fnum,d0       ; load from fnum
    addq.w    #1,d0         ; increment the counter
    move.w    d0,fnum       ; store it
    cmpi.w    #9,d0         ; have there been seven patterns
    beq       is_nine       ; go on if not
    jsr       drawfw       ; jump and XOR the fw explosion
    rts          ; leave

is_nine:
    move.w    numcount,d0   ; load numcount
    addq.w    #1,d0         ; increment counter
    move.w    d0,numcount   ; store it
    move.w    numexplode,d1 ; load explosion number
    cmpi.w    d0,d1         ; compare them
    beq       is_done       ; firework explosions finished!

    move.w    #1,fnum       ; reset fnum to 1

movea.l    #54bc,a0        ; pointer to system clock

    move.w    (a0),d0       ; get the value
    move.w    colormask,d1  ;
    and.w     d1,d0         ; mask to use maximum colors only
    cmpi.w    #0,d0
    bne       nzerol       ; insure plane one
    ori.w     #1,d0

nzerol:
    move.w    d0,fcolor     ; store in fcolor

    move.w    (a0),d0       ; get a number
    andi.w    #15,d0        ; and it to assure value in range
    move.w    (a0),d3       ; get another number
    andi.w    #7,d3         ; and it to assure value in range
    move.w    tx,d1         ; load x coordinate
    move.w    ty,d2         ; load y coordinate
    subi.w    #7,d1         ; subtract 4
    subi.w    #4,d2         ; subtract 4
    add.w     d0,d1         ; add the random value
    add.w     d3,d2         ; add the other random value
    move.w    d1,tx         ; store it
    move.w    d2,ty         ; store it

    jsr       drawfw       ; make the pattern
    rts          ; leave

is_done:
    move.w    #0,fnum       ; zero fnum

movea.l    #54bc,a0        ; pointer to system clock

    move.w    (a0),d0       ; get the value
    move.w    colormask,d1  ;
    and.w     d1,d0         ; mask to use maximum colors only
    cmpi.w    #0,d0
    bne       nzero2
    ori.w     #1,d0         ; insure plane one

```

```

nzero2:
    move.w d0,fcolor      ; store in fcolor

    move.w {a0},d4        ; now get another value
    andi.w #31,d4         ; and it to insure value is in range
    addi.w #40,d4         ; add 31
    move.w d4,fheight     ; store in fheight

    move.w {a0},d4        ; get another random number
    andi.w #3,d4          ; and it to assure value is in range
    addq.w #1,d4          ; add one
    move.w d4,numexplode  ; store the value

    move.w {a0},d4        ; get yet another value
    andi.w #3,d4          ; and it to assure value is in range
    move.w d4,trajectory  ; store in trajectory
    mulu  onethird,d4     ; adjust the starting point
    add.w  fromleft,d4    ; according to resolution

    move.w #199,ty        ; store the y coordinate starting point
    move.w #199,dl        ; put 199 in dl
    move.w d4,tx          ; store the x coordinate starting point
    move.w d4,d2          ; store x value

makepixel:
    move.w fcolor,d0      ; move the color into d0
    jsr  pixel            ; goto the pixel routine
    rts                  ; leave

continue:
    move.w vblanks,d0     ; get the vblanks counter value
    addq.w #1,d0          ; increment the counter
    move.w d0,vblanks     ; store the vblanks counter value
    cmpi.w #32000,d0      ; see if the appropriate time has passed
    bne  gohome           ; otherwise skip it

    move.l newscreen,d0   ; get the screen address
    movea.l d0,a0         ; move the newscreen address into a0
    move.w #0,d0          ; set the loop counter to zero

clearit:
    move.l #0,{a0}+       ; clear the screen
    addq.w #1,d0          ; increment loop counter
    cmpi.w #8000,d0       ; have 8000 longs been cleared?
    bne  clearit          ; otherwise do some more

    movea.l #544e,a0      ; set to _vbas_ad - contains the screen address
    move.l {a0},oldscreen ; get the old screen location

    movea.l #ffff8201,a0  ; set to the Physbase location
    movea.l newscreen,a1  ; move the address of newscreen into a1
    adda.l #1,a1          ; increment the pointer to get correct byte
    move.b {a1},{a0}       ; move the byte to the Physbase location
    adda.l #2,a0          ; increment to the other byte location
    adda.l #1,a1          ; add 1 to get the correct byte of newscreen
    move.b {a1},{a0}       ; move the second byte

    movea.l #5ff8260,a0   ; set to ff8260 - contains the resolution
    move.b {a0},d0        ; put resolution in d0
    andi.w #50003,d0      ; mask off any rubbish
    move.w d0,rez         ; store it in 'rez'
    move.w #0,oldrez      ; store '0' in oldrez

    cmpi.w #1,d0          ; see if we are in medium resolution
    bne  dontchange       ; if not, don't attempt to change res
    movea.l #5ff8260,a0   ; set to the hardware resolution location
    move.w {a0},d1        ; get the old resolution value
    move.w d1,oldrez      ; save it
    move.w #0,d0          ; move a zero into d0
    movea.l #544c,a0      ; set to the system variable location
    move.b d0,{a0}        ; store the new res
    movea.l #5ff8260,a0   ; set to the hardware location
    move.b d0,{a0}        ; store the new res
    move.w d0,rez         ; store the new res in 'rez'

dontchange:
    cmpi.w #0,d0          ; assign resolution dependant values
    bne  notlow
    move.w #2,vordbytes   ; bytes per 16 screen pixels
    move.w #5000f,colormask ; number of colors to be used
    move.w #80,onethird   ; horizontal distance (c. one third of screen)
    move.w #40,fromleft   ; value to start from left of screen
    move.w #1,xadjust     ; x factor
    bra  gotrez

```

```

notlow:
    move.w #2,wordbytes ; bytes per 16 screen pixels
    move.w #0,colormask ; number of colors to be used
    move.w #160,onethird ; horizontal distance (c. one third of screen)
    move.w #40,fromleft ; value to start from left of screen
    move.w #2,xadjust ; x factor

getrez:
    movea.l #oldpalette,a1 ; set destination of palette copy
    movea.l #fff8240,a0 ; set source of palette copy
    jsr getsetpalette ; jump and do the copy
    movea.l #newpalette,a0 ; set source of palette copy
    movea.l #fff8240,a1 ; set destination of palette copy
    jsr getsetpalette ; jump and do the copy

    move.w #1,vbiflag

    move.w #0,countfw ; zero the counter
    move.w #0,d7
    move.w #0,fnum
    move.w #0,numcount

four_fw:
    movea.l #54bc,a0 ; pointer to system clock
    move.w (a0),d0 ; get the value
    move.w colormask,d1 ; mask to use maximum colors only
    and.w d1,d0
    cmpi.w #0,d0
    bne nzero3
    ori.w #1,d0 ; insure plane one

nzero3:
    move.w d0,fcolor ; store in fcolor

    move.w d7,d4
    addq.w #1,d4
    mulu #10,d4
    addi.w #40,d4
    move.w d4,fheight ; store in fheight

    move.w (a0),d4 ; get another random number
    andi.w #3,d4 ; and it to assure value is in range
    addq.w #1,d4 ; add one
    move.w d4,numexplode ; store the value

    move.w d7,d4
    move.w d4,trajectory ; store in trajectory
    mulu onethird,d4 ; adjust the starting point
    add.w fromleft,d4 ; according to resolution

    move.w #199,ty ; store the y coordinate starting point
    move.w #199,d1 ; put 199 in d1
    move.w d4,tx ; store the x coordinate starting point
    move.w d4,d2 ; store x value

    move.w countfw,d7

    addq.w #1,d7
    move.w d7,countfw
    cmpi.w #1,d7
    bne pastfw1
    movea.l #fcolor,a0
    movea.l #fw2,a1
    jsr assign
    move.w fcolor,d0
    move.w ty,d1
    move.w tx,d2
    jsr pixel ; make the pixel
    bra four_fw

* Generate four sets of random firework data

pastfw1:
    cmpi.w #2,d7
    bne pastfw2
    movea.l #fcolor,a0
    movea.l #fw2,a1
    jsr assign
    move.w fcolor,d0
    move.w ty,d1
    move.w tx,d2
    jsr pixel ; make the pixel
    bra four_fw

pastfw2:
    cmpi.w #3,d7
    bne pastfw3

```

Public Domain Software

#57 - Tease Me Adult Animation (Color Only)
 #145 - Five Children's Programs (Color Only)
 #352 - Lost Treasure (Lode Runner Clone) - Color
 #393/394/533 - PrintMaster Graphics
 #400 - 7 Disk Labeling Programs (w/100 Labels \$6.95)
 #443 - Internet RAM Baby (RAM Disk/Print Spooler)
 #456 - Bolo Breakout Game from Germany (1 Meg)
 #470 - Two Virus Killer Utilities, Database and more
 #475 - Werry's House of Horror (Adult Game, Color)
 #500/600 - Publishing Partner Fonts
 #511 - Dungeon Master Maps for Levels 1-7
 #512 - Dungeon Master Hint/Character
 #555 - The Assistant Chef - Electronic Cookbook
 #557 - Children's Programs (Color Only)
 #567/728 - Accessories
 #575 - Sheet V2.0 - Shareware Spreadsheet
 #580 - Pac Man, Hangman and 5 others (Color Only)
 #589 - PageStream Fonts, Font Converter
 #655 - ST Writer V3.0 w/Spell V2.8
 #690 - Opus V2.10 GEM Spreadsheet (1 Meg/DBL)
 #700 - ST Xplorer V2.4 (Xformer Cable \$19.95)
 #721 - Drawing Programs

Introductory Offer - Above Disks Just
\$2.99 Each
 Call or Write for FREE Catalog
(800) 347-6760

BRE Software Dept. STL
 352 W. Bedford, Suite 104
 Fresno, CA 93711

Customer Service (209) 432-3072
 Shipping, 10-20 Business Days



CIRCLE #102 ON READER SERVICE CARD.

```

movea.l #fcolor,a0
movea.l #fw3,a1
jsr assign
move.w fcolor,d0
move.w t9,d1
move.w tx,d2
jsr pixel ; make the pixel
bra four_fw

pastfw3:
cmpl.w #4,d7
bne gohome
movea.l #fcolor,a0
movea.l #fw4,a1
jsr assign
move.w fcolor,d0
move.w t9,d1
move.w tx,d2
jsr pixel ; make the pixel

gohome: rts ; leave

*****
* Here is the keyboard patch *
*****

trapper:
move.w #0,vblanks ; reset the vbi counter to zero
move.l a0-a1/d0/d7,-(sp) ; save addresses and loop counter
move.w vbiflag,d0 ; get the value in the vbi flag
cmpl.w #1,d0 ; see if it is one (enabled)
bne getlost ; if not leave...

movea.l #ffff8201,a0
movea.l #oldscreen,a1
adda.l #1,a1
move.b (a1),(a0)
adda.l #2,a0
adda.l #1,a1
move.b (a1),(a0)

move.w #0,fnum ; zero fnum in case of fireworks going off

movea.l #fff8240,a1 ; set destination of palette copy
movea.l #oldpalette,a0 ; set source of palette copy
jsr getsetpalette ; jump and do the copy

move.w #0,vbiflag ; set the vbi flag to zero

move.w oldrez,d0
cmpl.w #0,d0
beq getlost
move.w #1,d0
movea.l #544c,a0
move.b d0,(a0)
movea.l #fff8250,a0
move.b d0,(a0)

getlost:
move.l (sp)+,a0-a1/d0/d7 ; restore addresses and loop counter
insert:
jmp $11111111 ; this address is changed upon program init

*****
* This routine XORs a pixel on the screen *
*****

pixel:
* Pixel color in d0
* y coordinate in d1
* x coordinate in d2
* use a3 for screen address
* use a4 for bit mask

movea.l newscreen,a3 ; put the screen address in a3
mulu #16,d1 ; multiply the y coordinate by 16
andi.l #5000ffff,d1
adda.l d1,a3 ; add the y value to a3
andi.l #5000ffff,d2
divu #16,d2 ; divide the x by 16 to get the number of words
move.l d2,d1 ; store the result in order to ascertain the remainder
andi.l #5000ffff,d2

```



```

        mulu    wordbytes,d2    ; multiply by wordbytes to find the correct byte offset
        andi.l  #$0000ffff,d2
        adda.l  d2,a3           ; add the byte offset to a3
        lsr.l   #8,d1           ; shift 8
        lsr.l   #7,d1           ; shift 7
        andi.l  #$000000fe,d1   ; offset in d1

        movea.l #pixbits,a4     ; put the pixbit address in a4
        adda.w  d1,a4           ; add the correct offset to a4
        move.w  (a4),d3         ; move the data into d3

* Move the pixel data is in d3 and the screen address is in a3

        btst   #0,d0
        beq    secondplane     ; if not, skip to the second plane

        move.w  (a3),d2         ; get the screen byte
        eor.w   d3,d2           ; XOR it with the data
        move.w  d2,(a3)         ; put it back on the screen

secondplane:
        adda.w  #2,a3           ; add the correct number of bytes
        btst   #1,d0
        beq    thirdplane      ; if not, skip to third plane

        move.w  (a3),d2         ; get the screen byte
        eor.w   d3,d2           ; XOR it with the data
        move.w  d2,(a3)         ; put it back on the screen

thirdplane:
        adda.w  #2,a3           ; add the correct number of bytes
        btst   #2,d0
        beq    fourthplane     ; if not, skip to the fourth plane

        move.w  (a3),d2         ; get the screen byte
        eor.w   d3,d2           ; XOR it with the data
        move.w  d2,(a3)         ; put it back on the screen

fourthplane:
        btst   #3,d0
        beq    lastplane       ; if not then leave

        adda.w  #2,a3           ; add the correct number of bytes
        move.w  (a3),d2         ; get the screen byte
        eor.w   d3,d2           ; XOR it with the data
        move.w  d2,(a3)         ; put it back on the screen

lastplane:
        rts                    ; leave the routine

*****
*      Get/Set current palette (must be in Supervisor)      *
*****
getsetpalette:
        move.w  #0,d7           ; zero the counter

gsploop:move.w  (a0)+,(a1)+      ; move the data
        addq.w  #1,d7           ; increment the counter
        cmpi.w  #16,d7          ; have you copied 16?
        bne     gsploop         ; if not, do some more

        rts                    ; leave subroutine

*****
*      Draw a firework explosion part                        *
*****
draufw:
        move.w  fcolor,d0       ; get the firework color
        move.w  #0,d7           ; zero the counter
        movea.l #fpattern,a5    ; get the address of fpattern
        move.w  fnum,d5         ; load the firework number
        move.w  d5,d5           ; save the firework number in d5 for future use
        mulu    #3,d6           ; figure out the radius

do_dots:
        move.w  tx,d2           ; get the x coordinate
        sub.w   d6,d2           ; adjust to top left of firework pattern
        move.w  xadjust,d4      ; load xadjust
        cmpi.w  #2,d4           ; see if it is 2
        bne     xa_not_2        ; go on otherwise
        sub.w   d6,d2           ; adjust additional amount
        xa_not_2:

```

```

move.w    ty,d1          ; get the y coordinate
sub.w     d6,d1          ; adjust to top left of firework pattern
move.w    (a5)+,d4        ; get the x pattern element
mulu      d5,d4          ; multiply according to the size of the fw
mulu      xadjust,d4      ; resolution factor
add.w     d4,d2           ; add the x position to the top left of pattern
move.w    (a5)+,d4        ; get the y pattern element
mulu      d5,d4          ; multiply according to the size of the fw
add.w     d4,d1           ; add the y position to the top left of pattern
addq.w    #1,d7           ; increment the counter
jsr       pixel          ; now, make the pixel
cmpl.w    #8,d7           ; see if you have done the 8 dots
bne       do_dots         ; do some more if not

rts                ; leave the routine

```

```

*****
*           Move the specified firework data to the work area           *
*****

```

```

assign:
move.w    #0,d0

do_more:
move.w    (a0)+,(a1)+
addq.w    #1,d0
cmpl.w    #10,d0
bne       do_more

rts

```

```

*****
*           Data Storage Segment                                         *
*****

```

```

                .even
                .data

pixbits:        .dc.w    $0000,$0000,$2000,$1000,$0000,$0400,$0200,$0100
                .dc.w    $0080,$0040,$0020,$0010,$0000,$0004,$0002,$0001

newpalette:     .dc.w    $0000,$0070,$0700,$0700,$0770,$0707,$0077,$0777
                .dc.w    $0770,$0704,$0074,$0740,$0704,$0740,$0400,$0700

fpattern:       .dc.w    $0000,$0003,$0001,$0001,$0003,$0000,$0005,$0001
                .dc.w    $0006,$0003,$0005,$0005,$0003,$0006,$0001,$0005

fw1:            .dc.w    $0000,$0000,$0000,$0000,$0000,$0000
                .dc.w    $0000,$0000,$0000,$0000

fw2:            .dc.w    $0000,$0000,$0000,$0000,$0000,$0000
                .dc.w    $0000,$0000,$0000,$0000

fw3:            .dc.w    $0000,$0000,$0000,$0000,$0000,$0000
                .dc.w    $0000,$0000,$0000,$0000

fw4:            .dc.w    $0000,$0000,$0000,$0000,$0000,$0000
                .dc.w    $0000,$0000,$0000,$0000

```

```

*****
* Block Storage Segment

```

```

                .bss

rez:           .ds.w    1          ; current resolution
oldrez:        .ds.w    1
oldscreen:     .ds.l    1          ; old screen pointer
keyvector:     .ds.l    1          ; keyboard vector
oldpalette:    .ds.w    16
wordbytes:     .ds.w    1          ; bytes per screen word
colormask:     .ds.w    1
onethird:      .ds.w    1
fromleft:      .ds.w    1
xadjust:       .ds.l    1
newem:         .ds.l    8200
newscreen:     .ds.l    1
vblanks:       .ds.w    1          ; vblank counter
countfw:       .ds.w    1          ; status of the screen saver
vblflag:       .ds.w    1          ; 0=not enabled
                                ; 1=enabled

fcolor:        .ds.w    1
numexplode:    .ds.w    1
trajectory:     .ds.w    1
fheight:       .ds.w    1
ty:            .ds.w    1
tx:            .ds.w    1
fnum:          .ds.w    1
numcount:      .ds.w    1
fwvblanks:     .ds.w    1

```

```

                .ds.l    255        ; the stack can grow down 1K before
ustk:          .ds.l    1          ; it starts to eat the program!!
                .end

```

THE ULTIMATE SCREEN SAVER

END



4049-51 MARSHALL RD. • KETTERING, OHIO 45429

CIRCLE #103 ON READER SERVICE CARD.

the

animati

BY MAURICE MOLYNEAUX

the time for just conceptualizing is over! Time to take all those design sketches, storyboards and other preproduction material and put it to work. The time has come to sit down at your ST and start animating.

We interrupt this article. . .

Ah, but before we begin working in earnest, now's a good time to tell you about a neat little tool for getting a different perspective on your design.

In traditional (full) cel animation, particularly of the pose-to-pose variety (see last issue), when the key poses for a sequence have been drawn, the existing work may be committed to film to create what is known as a "pose reel." A pose reel checks the action before the dirty work of the actual animation begins. How? Each pose is photographed, not for only the frame or two for which it would appear in the final animation, but for as many frames as necessary until the next pose is struck. For instance, if Daffy Duck is looking to the left in the first pose, and the next pose has him looking to the

right, and there will be twenty positions in-between those poses, the first pose is photographed for the full twenty frames, then the next is photographed for as many frames as will exist between it and the following one. This can then be viewed to see how the whole sequence of action will look and how far apart (timingwise) all the key actions are. In this manner, the animators can roughly determine if the action happens too fast or too slow, or if there are other problems.

But how can you do this on your ST? Quite simple, really. You generate graphics of each key frame in your storyboard, set them up so that they can be cycled through with a key press and videotape your ST's output (videotaping will be covered at the end of the series) as you advance through these images.

For example, with the music video I am

working on I could create a pose reel either by using a video digitizer or image scanner to "grab" the individual images from the actual storyboard, or I could draw rough likenesses of these actions in a paint program. Once I had a series of these, I would somehow place them in an order that would allow me to step from one to the next almost instantaneously. A slideshow program may be able to do this (you have to make sure the pictures will cycle in the correct order), but you'll need to have it loading the images from either a fast hard disk or a RAM disk to be able to have the pictures advance when you want them to. Personally, I assemble all the pictures into a .SEQ file in *Cyber Paint* so I can advance through them forwards or backwards with the single-frame advance/rewind keys.

Now, making a pose reel may not seem very useful to you, but in some cases it's an extremely helpful tool. In my case, I am going to be timing my animation to a pre-existing piece of music. My timing *must* match the song. By creating a pose reel, I can see if what I'm attempting will work in the time the song permits.

As to how I'll get these pictures on tape in time with the music, that's simple. Hook a video-out from the ST to the VCR's video-in (again, I'll cover this in a later article), and the audio-out from my stereo system to the VCR's audio-in. I get the sequence of pose-reel images into the computer's memory, then start the VCR recording. Moments later, I start playing the music/soundtrack. As the music plays, I will advance through the images at the planned spots in the score. When I'm fin-

on stand:

ished, I can study the videotape repeatedly (and also show it to colleagues for criticism and suggestions) to see if any potential problems crop up. If nothing else, this method gives a much clearer idea of what your final product may be like. Often, you'll find it's not quite what you expected!

How you create the raw images for a pose reel depends on how complex you want to get and what software you plan to use for the final animation. If you are using a paint-animation or cel-animation system (like *Cyber Paint* or *Film Director*, respectively), you can just draw (or digitize) pictures to match those in your storyboard. However, if you're using *CAD-3D*, you may want to build simple "test" objects and generate superviews (and save them to disk!) of them in positions that match your storyboard. Once you've done that, you can make a pose reel out of those images.

■ Under construction

Once the design phase is completed, it's time to start creating the necessary elements for your animation. If you're using *Film Director*, you'll need to start drawing "shape tables," which are screens of objects you'll cut out and use as cels. If you're using *CAD-3D*, you'll need to start constructing the objects you will be animating (unless, of course, you are going to have a *Cyber Control* program "create" those objects on the fly). However, if you're going to animate "straight ahead" with something like *Cyber Paint* (or the upcoming *DaVinci* from Artisoft), or using the polygonal/metamorphic features of *Animator ST*, chances are you won't be do-

ing much pre-animation drawing.

Here are a few pointers for this stage:

If you're drawing shape tables, try cramming as many elements as you can on a single page, but try to keep them organized. Group all the hands together, the same with eyes, limbs, and so forth. Also, if you have a series of parts designed to be used in a particular movement sequence (such as leg positions for a walk cycle), try to arrange those in a row so that you never have any trouble remembering the sequence in which you intend to use them.

Another tip: Since every cel-animation system I'm aware of "clips" the individual elements out as rectangular objects (with the background color transparent), you have to make sure that no two objects creep into the same "box." I suggest drawing a thin rectangular frame around each object. This not only keeps you from overlapping any two objects, but it also makes it easy for you to clip out the cels in the animation utility because you can see exactly where the edges of the cels' area are.

For animations using 3-D objects, you should endeavor to keep your objects as simple as possible. I don't mean simple in appearance, but in construction. The more complex an object is, the more faces (the flat segments that make up the object) it will contain. The more faces there are, the longer it will take *CAD-3D* (or any other program that can do this kind of animation) to manipulate the objects and generate frames of them. If you are going to use *CAD-3D* and you need reasonably complex objects, you should buy *Cyber Sculpt*. This product allows you to have

tremendous control over the 3-D objects, and even clean up messy joints that an object originally created in *CAD-3D* might contain.

■ Making a scene

Just about every animation program for the ST will let you use a full-screen image for a background. In *Film Director* you merely load the picture into one of the shape table screens, cut it out as a single large cel and place it on one of the "stages." In *Cyber Paint* you can "underlay" a picture under a selected series of frames. You can load a picture as a background for *CAD-3D* animations (only useful if you don't plan to move the "cameras") as well. You may want to create such background scenery (at least in rough form) before beginning to animate, because you may have to move a character or object relative to something in the background, like a mailbox, space station, or what have you. You can't accurately animate an object to match such a point unless you have some kind of background so you can see exactly where that object is while you're actually animating.

Fortunately, you can usually replace the background behind an animation even after you've completed it (unless you've created a delta or sequence animation with a background already present, in which case removing the existing background could prove troublesome). Because of this, you could merely draw some rough outlines showing where elements will be in the background, then go back, after animating the scene, to draw an appropriately detailed setting. You might want to do this so you can get right to the

animation, or if you're not sure precisely what kind of background you need, or for some other equally good reason.

■ The pencil test

I am now assuming that you've completed any necessary creation of graphics objects for your animation. The next thing to do is attempt to animate a small portion of it in order to see if it works. In cel-animation circles, the first test animation consists of photographing the actual animator's pencil drawings to see how the scene or character looks. This is known as a "pencil test," and you can consider this step the digital equivalent.

At this point you should pick a shot (if your animation has more than a single one) that will give you a chance to see if the elements you've created will work. For example, when I draw the parts for a cartoon character the first thing I do is try to animate it walking, then turning. This is a great test for characters, because if something doesn't look right, you can usually spot it immediately. Basically, when test-animating a character, you should either pick a scene in which the character has to go through a broad range of actions, or you should consider simply animating a disposable "screen test" to see if everything works.

As to CAD-3D-type animations, you'll probably want to generate a rough of what the finished shot will look like. If the objects you've created for the scene are complex, you should consider replacing them with very simple "stand-in" objects during these tests, just to get everything up to speed. This is a good idea because complex objects take a while to move, and a nice final-draft mode (CAD-3D 2.0 only) superview can take a while to generate as well. Just last night I created a 200-frame fly-by of the *USS Enterprise*. The object has over 4,000 faces, and each superview took two minutes to generate. All in all, it took over six hours to generate these frames (I let *Cyber Control* do this job while I was sleeping), and all I was doing was moving the camera in a straight line past the ship. Had I actually had the ship moving relative to the light sources or done more complex object manipulation, it could have taken as much as twice that long! (I know some people who've had CAD-3D rendering for over two days, nonstop.)

In either case, you should now have an idea of how your characters/objects move. As always, attempt to find potential trouble spots and fix them before beginning your animating in earnest.

■ And action!

At this stage, you should be ready to get

under way with the real thing. There are many different approaches to going about the actual production phase, but here are some general pointers that you might find helpful:

Have some scratch paper and a pencil on hand. As you're animating, you may find that you hadn't thought out some actions as clearly as you should have, or you might want to try a different or new approach. If you have to experiment, sketch as necessary. The computer is a great tool for animation, but it isn't yet as fast or effective as pencil and paper for roughing out visual action.

Keep your storyboards on hand at all times and refer to them often. If your animation is at all complicated, it's easy to forget some small actions you had planned if you begin animating without them. It's also easy, in the heat of the process, to lose sight of your ultimate goal. For instance, you might get an idea to have a character do some additional bit of business at some point, but in the process unintentionally draw the attention of your audience away from where you're supposed to be leading. It's often a tough call to determine when your changes hinder rather than help. You'll have to try to keep an open mind and be willing to say, "I like that, but it's doing more harm than good."

On the other hand, you should use your storyboard as a guideline, an outline of what you're trying to do. Do not consider it a rigid instruction manual. Just because you don't follow the directions to the letter, doesn't mean you're going to break anything. If, in the middle of animating you find yourself coming up with new ideas, don't ignore them. If you can see how those ideas can help your work, by all means try to integrate them. This doesn't mean those new ideas won't have any effect in the long haul, because they are likely to. You just may have to make other alterations to get your overall goal back on track, or be willing to let the work become whatever it is changing into (hopefully, something good).

Small changes in a specific bit of business will rarely make much difference in the long run. A lot of the funnier bits of business I put Megabit Mouse through in the *Art & Film Director* sales video were not storyboarded. I thought of them as I was animating. As they were mostly just embellishments on the planned action, their effect was to add to Megabit's character and make the scenes funnier without changing the overall thrust of the video. They made little difference in the long

haul, other than making the animation more entertaining (and that's no small thing).

If you are working with a program where you can play back your animation as you work on it, do so frequently. Loop through each new action and view everything you've done thus far every now and again. As the scene starts coming together and you can see it actually moving, you may find yourself getting new ideas or spotting mistakes you'll want to fix. Also, seeing the flow of the motion can often show you an alternative way of getting from one action to another, particularly when animating characters. Let's say you have a cat that runs into the scene, screeches to a halt and looks into a trash can. From the trash can he is supposed to bring his head up and swivel it around to look behind him. If you just determine the shortest path between these poses and animate according to that, you may find it doesn't look right. Something about the action is unnatural. When you play back the part of the scene you have completed, you can see the flow of the action thus far and try to make the next action follow that flow. I can't really explain why this works, but if you focus on seeing paths of action you will probably catch on to what I'm talking about here.

Oh, and save your work frequently. Even if you know better, it's easy to work and work and work without saving anything, and then have the program crash or your ST lock up. It goes without saying that if you've made 25 major changes to your animation without saving, you'll have some kind of crash that will send all your hard work to electron heaven. Be safe! Save your work at least every 15 minutes and make backup copies. A friend of mine lost a lot of his artwork because a bunch of his disks went bad, and he didn't have backup copies of any of them.

■ Next time

That's all for this time. Next time the topic will be animation techniques. From squash and stretch and motion blurs to slow ins and outs, I'll be giving you a lot of tips on how to make your animations smoother and better looking. ■

Blissfully ignorant of the realities of time and space and plain old common sense, Maurice Molyneux hopes someone will someday discover "retroactive reincarnation" so that when he dies he can come back in a previous life as animation director Chuck Jones. His greatest fear is to come back as Wile E. Coyote, and in the process have to learn some humility.



12 Issues \$28

\$19 OFF THE COVER PRICE

12 Issues with Disk \$79

NEW LOWER PRICE

BREAK AWAY FROM THE PACK



The world of ATARI-ST continues to grow by leaps and bounds, and ST-LOG is there every step of the way! We stand apart from the competition by offering more color, comprehensive reviews and in-depth features. **SUBSCRIBE NOW!**

☐ **12 Issues \$28**
MCUWW



☐ **12 Issues with Disk \$79**
DCUWW

☐ PAYMENT ENCLOSED ☐ BILL ME
CHARGE MY: ☐ VISA ☐ MASTERCARD

CARD # _____ EXP _____ SIGNATURE _____

NAME _____

ADDRESS _____

CITY _____ STATE _____ ZIP _____

MARKS CHECK PAYABLE TO L.F.P., INC., P.O. Box 146928, N. Hollywood, CA 91618. Offer Expires Nov. 29, 1989

MAGNIWRITER

Bruce D. Noonan, M.D.

NOTE: Due to the large size of this program, it is available only on this month's disk version or from the databases of the ST/LOG ST users' group on DELPHI.

Most of us take our vision for granted. Sure, some of us wear glasses or contacts, but most people can have vision corrected to 20/20. For 11.4 million visually impaired Americans, however, just getting around the house is a major chore fraught with many obstacles and possible dangers. These individuals are often unemployable because they are unable to drive or see well enough to board a bus. Surprisingly enough, many of these same individuals have turned to computing as a means to make a living—specifically, word processing.

Most large-print word processors require the use of an IBM PC with special hardware and software. Approximately \$5000 was required to obtain the setup. *Magnewriter* and the ST present a more economical alternative. With the addition of a modem, one could produce the text on the word processor and transmit it to its destination for printing over normal phone lines. Needless to say, for low-vision individuals this can be the answer to their employment difficulties.

For several years now, I have been working on *ST Writer*, Atari's own word processor, distributed free with computers back in 1985, prior to the arrival of *1st Word*. Although *ST Writer* had lots of bugs, I liked it because it was nearly identical to *Atariwriter* for the 8-bit machines and I was already familiar with the formatting commands. After coaxing Atari to release the source code to me, I fixed the bugs and even made it into a GEM half-breed. Now called *ST Writer Elite*, it can either be used with or without the mouse and drop-down menus.



Since I am an ophthalmologist (eye surgeon), I toyed with the idea of creating a large-print word processor for the ST. I had written a program for the 8-bit machine called *Magnifile Text-Reader*, which read in *Atariwriter* or any ASCII files and printed them in large yellow letters on a black background (high contrast). It had search routines, line isolation and two sizes of text (large and extra-large). Currently, that program is available both on CompuServe and GENie.

I looked at *ST Writer* and thought, "Why can't I make those letters large?" I toyed around with the source code, and *Magnewriter ST* was the result. It has all the features of *ST Writer* except for GEM (I couldn't figure a way to make all the menus and dialog boxes, etc., with large print). But I did add a few features, in-

cluding a "Save ASCII" ability. Files thus saved are not saved in *ST Writer* format, contain no formatting or printing controls but are compatible with nearly all word processors. The only margin setting that can be used with these files is the right margin. This can vary from up to 160 clear up to 196, depending on whether you are using an 80-column printer driver or a 132-column printer driver.

You can create a printer driver by loading in CONFIG.TXT into *Magnewriter*, entering your control codes for your specific printer, save the file (not Save ASCII), exit the program by typing "Q" at the menu screen, then run CONFIG.TOS from the desktop. The program reads the CONFIG.TXT file and writes a file called XYZXX.DAT—the printer driver file. This file must be in the same directory as

MGWRITER.PRGM in order to be loaded with the program. If you already have a printer driver for *ST Writer Elite*, it will work for *Magnewriter ST* as well.

Magnewriter ST can run in all three resolutions but has the largest text in low resolution, ten lines and 20 columns. Medium resolution allows 40 columns and ten lines, and high allows 40 columns and 20 lines. In order to use all three resolutions, however, the program had to be geared to the resolution with the least number of columns available (low). A problem with low resolution is that all prompts and responses must total 19 characters or less, plus a carriage return. Thus, many abbreviations are used, as well as chopped-up phrases.

If you are already familiar with *ST Writer*, you will not have any trouble. If you are using folders, you may not be able to get a drive specifier, folder and filename on a single line. If you must use folders, name them with a single letter and use fewer letters in filenames. Otherwise, save your file to the main directory and do your housekeeping from the GEM desktop.

For those interested in how the large letters were obtained, I used the 8x16 character set in the *ST*. For medium and low resolution, usually the 8x8 set is used. However, by designing a custom character set, you can use any of the three available or design one of your own. The third set, 6x6, is used for the icons on the desktop and by various dialog boxes in application programs.

Since the 8x16 set was already twice

the normal height, all that was needed was to make each letter double-wide. Although the line-A variables allow access to the font variables, one is restricted from creating a character wider than eight bits (one byte) when using *Beamout* () to print to the screen. What I was able to do, however, was create two character sets, one with the left eight bits of the letter, and the second with the right eight bits of the letter. I then printed half from the first character set, switched to the second character set immediately and printed the other half of the character. The *ST* microprocessor is so fast, it appears that both halves of each letter are being printed as a single letter.

You may notice that the cursor is only one-half the width of the letters—it also is restricted to eight bits in width. Oh yes, before I forget, the characters are 20 pixels high, not 16. This allows four pixels of blank space between lines. Fortunately, the width restrictions on characters did not apply to their heights as well.

You may be thinking, "Can I print out text in large letters on my printer with *Magnewriter*?" The answer is "yes and no." It depends on the capability of your printer. My Star NB24-15 has the capability of printing quad-sized letters if I send it the command to do so. With *Magnewriter*, the proper codes for double-sized letters at the top of the file will be:

Control-O 27

Control-O 104

Control-O 1

Just remember that when characters are double-wide and double-high, you can get only one-fourth the usual amount of text on a page. You will need to readjust your top, bottom, left, right and page-length settings to one-half their usual parameters:

Control-B 6

Control-T 6

Control-L 5

Control-R 35

Control-Y 66

It is always a good idea to save your file frequently when editing. If you have a particular preference for margin and style settings in the format line at the top, you can set them the way you like, then save the file. Every time you want to use those settings, merely load in the saved file with the format line of your choice.

Since Atari still legally holds all rights to the source code, this program cannot be sold and is in the public domain. The source code remains the property of Atari Corp., and therefore is not included on the disk. You will, however, find several

other files. *MAGNIMANT.TXT* is the tutorial and manual for *Magnewriter ST*. *QUIKREFTXT* is an abbreviated manual. In order to print these or to view them, you must first run *Magnewriter ST*, then use Load to load the file. Type "P" for Print, then decide if it is to be printed to the screen or to the printer when prompted. If you print to the spooler, formatted text and printer controls will be written to a disk file. Spooler files can be printed directly via the Print options from the desktop by simply clicking on the file.

MAGCODESTXT is a file that shows which keys produce ASCII characters greater than 127 using the Alternate-X toggle. Some foreign vowels and characters can be produced using the Control-Clr-Home "deadkey" toggle. With this, you can produce an "umlaut o," for example, by first typing the double-quote key, then the "o." Note that nothing will appear until the "o" is typed. If you were to type the double-quote twice in succession with the deadkey toggle on, you would get one double-quote character on the screen. The little letters on the right side in the box at the bottom indicate the various states of the toggles:

D = Deadkey (Control-Clr-Home)

A = Special Characters (Alternate-X)

I = Insert Mode (Insert)

T = Typeover Mode (Insert)

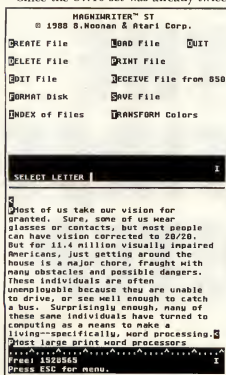
C = (Capslock)

For further help, see the manual and/or the file *HELP.TXT*. *Magnewriter ST* can also print out a function key template for you to place above your function key row. Load the file *TEMPLATE.TXT* and print it. Your printer needs to know the Epson control codes for condensed text to print it correctly.

I hope you enjoy *Magnewriter ST*. Perhaps you can get the kids interested in using a word processor with the large letters.



Dr. Noonan is an ophthalmic surgeon in private practice. He has been continually updating and improving ST Writer since 1986. He has also written a medical program for the ST that calculates intraocular lens powers for artificial lens transplants in cataract surgery patients.



Where has the time gone? According to the cover, this is the 35th issue of SFLOG. That's just one issue shy of three years! But wait. The first SFLOG magazine hit the stands at the end of 1986. "That's not three years!" you say. That's because the first issue was actually the tenth. SFLOG appeared as an "insert" in ANALOG Computing for nine issues prior to the big split. So, technically, this is the 26th SFLOG magazine.

Twenty-six "real" issues, and *Step 1* has appeared in all but two of them (issues 10 and 22). In a way, *Step 1* has been a part of this magazine since the beginning. From my estimation, the only regular feature to appear in SFLOG more often than this column is *Ian's Quest*, coming in at twenty-five issues. But as far as total wordage goes, I can proudly say I've written more than any other SFLOG regular. *Step 1* has usually been the largest regular feature, and an approximate count puts the total (including this issue's column) at over 96,000 words, more words than appear in an average 200-page paperback novel.

But all good things must come to an end, and with this issue *Step 1* takes its final bow.

Why? There are several good reasons. First, in the course of twenty-three articles, I've covered everything from basic hardware and software to telecommunications and GDOS, and on the way taken a look at graphics, animation, utility programs and so forth. There's little "beginner" material left to cover.

The second reason is fatigue. Cranking out 4,000 words per month of beginner/ novice-level material is no easy trick, particularly when topics appropriate for your audience become harder and harder to find.

And no, this is not some overdramatic way of launching a *Step 2* column (the one that my friends joked I would eventually be forced to write). There won't be one.

This isn't to say I'm bowing out of the ANALOG/SFLOG family either. The only way you're getting rid of me any time soon is if I'm dragged off kicking and screaming. I'll continue writing about Atari computers, but no longer in the monthly capacity of this column.

But before I go on to the subject of this final *Step 1*, I'd like to say "you're welcome" to all the readers who've enjoyed and/or been helped by these articles. The most gratifying part of writing *Step 1* has

Step 1:

MAURICE MOLYNEAUX

been the response I've gotten from you readers. I've gotten quite a few "thank you" notes over the past two and a half years. Believe me, it's a great feeling to know you've been able to help so many people in even a small way.

So, I'd like to return the favor by saying "Thanks" to you readers. It's been fun.

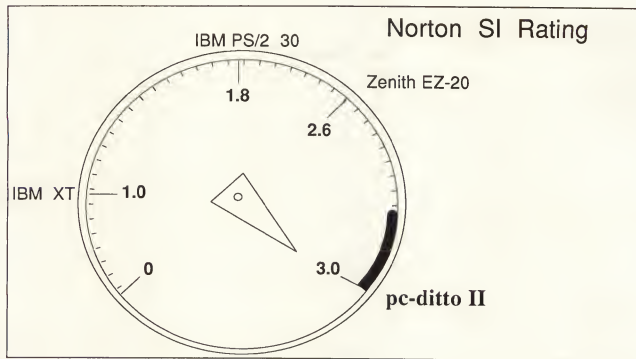
The ST world today

I decided to dig up the first few *Step 1*s just to see how much different things were then from now. In a way, I'm surprised at how little things have changed. The ST's market presence in the U.S. is still weak, a lot of the available software still has serious problems. Atari still hasn't shipped the CD-ROM. The blitter chip finally appeared in the Mega SIs, but plans to "retrofit" existing 520 and 1040 machines have been abandoned. No major hardware upgrades have appeared, like an ST with slots, a faster processor or improved graphics and sound. The AMY sound chip is as much a piece of vaporware as it was when the ST was first announced.

THE FIRST STEP

pc-ditto II

By Avant-Garde Systems



IBM XT COMPATIBLE !

IBM AT PERFORMANCE !

ATARI ST PRICE !

\$299.95

Now! Run the most popular IBM software on your Atari ST...

FAST !

See pc-ditto II at your local Atari dealer or write for free information.

Yes. Please send me more information !

Avant-Garde Systems, 381 Pablo Point Drive
Jacksonville, Florida 32225

Name

Address

City St Zip

Single-sided drives are still (unfortunately) with us, and people still complain about Atari's market and customer support.

Then again, we've gotten better hard disks, laser printers and multi-megabyte STs. Third-party peripherals, such as genlocks, big-screen monitors, color-palette booster boards and boards with faster microprocessors have also appeared. The ST is the #1 MIDI computer and a top seller in Europe. All fine and good, but not quite what we'd all hoped and expected.

And the users haven't changed much either. It seems the ST attracts a lot of first-time computer buyers who are attracted by its low price and ease of use. I imagine that if I had started writing *Step 1* with this issue instead of way back in late 1986, the response to it would have been about the same as it was then.

If one thing has changed, it is that it seems users—even new users—are more aware of what they want their computers to do. Desktop publishing, graphics and animation, etc., are now good reasons to purchase an ST. It proves what many of us suspected all along: The average person needs a good reason to buy a computer. Offering a low-priced machine does not ensure success. (Are you listening, Atari?) The one good thing about this is that users who buy a machine for a specific and useful task are less likely to be disappointed and badmouth it. This is because they have more realistic expectations than those who bought one because they expected the computer to "do something" for them. Which brings up the question of . . .

A white elephant

Why did you buy a computer? Think about that for a moment. Did you purchase it because you were interested in computers, or did you have a specific task in mind? I've seen many people buy computers to "balance my checkbook." *Puh-lease!* Spend a thousand dollars or more to do what you can do with a pocket calculator? Not so oddly, most people who purchased a computer for such a reason never use it for that task. Either it becomes an expensive video game or it collects dust in a dank closet somewhere in the bowels of their homes.

Why? Because more often than not, these people expected the computer to do something for them. The computer would make them more productive, help them

manage their time, make them better writers, musicians or whatever. But in 99 percent of all cases, it does none of these things; the computer doesn't live up to the expectations it was purchased under. It becomes an expensive and essentially useless object—a high-tech white elephant.

But what went wrong? Was it that the purchaser's expectations were too high, or that the computer was simply incapable of doing the tasks expected of it? Both, and maybe neither.

The "Enabler"

As I stated last issue, a computer by itself won't do a thing for you. A present-day personal computer is essentially a complex and flexible programmable calculator—a stupid piece of hardware that processes 1s and 0s. Give it some instructions, and it does precisely what it's told. No more, no less. It can't think, it can't help you when you're in trouble. It won't teach you to be a better anything. It's just a multimillion-bit number cruncher with a fancy name.

If that's so, why do computers have such a reputation as machines that will change your life, even though the majority of buyers find their lives and working habits changed not one iota?

I think it's because there is some truth to the notion that computers can change your life, but the problem stems from the fact that the statement says one thing, yet means another. It says, "a computer can change your life," which implies that the subject, the computer, is what's making the change. The statement should read, "You can use a computer to change your life." See the difference? In the latter statement it is *you* who are the subject, *you* making the changes. The computer itself does nothing; rather, *how you use the computer* makes the key difference.

Let's say you bought a copy of Suzan Haden Elgin's *The Gentle Art of Verbal Self-Defense*. (This is not an endorsement of the book; I'm just using it as an example.) Putting that book in your bookshelf isn't going to do anything for you. Reading the words within it won't help either. However, if you use the knowledge you've gleaned from reading the book, you may find that you can hold your own in arguments much better than you have in the past.

You purchase an ST and a copy of *WordPerfect*. Putting that software on the shelf or using it to print recipe cards is not going to make you a better writer.

Writing a story and running the spell-check on it won't do it either. To be a better writer, you have to critically and honestly evaluate your work, edit and rewrite, and never settle for what you have as "okay." You always have to look at how to make it better.

Using the computer to make you a better writer means taking advantage of its ability to make easier the task of typing the words. You don't have to retype pages to insert a missed paragraph or change a name. You don't have to watch out for the right margin or getting to the bottom of the page. You don't have to spend time worrying about typographical mistakes. You can fix and change these at any time. The distracting aspects of the writing process have been minimized so you can concentrate on the actual *act* of writing. You can work more quickly and easily. The computer and software have taken away some of the burdens, allowing you to concentrate on the task at hand and also permitting you to freely experiment. You can easily try adding a phrase, deleting a line or moving a paragraph, and if you don't like it, you can have the text back in its original form as quickly as you made the changes.

Try that with a typewriter.

So a word processor won't make you a writer, a paint program won't make you an artist and a desktop-publishing utility won't make you a typesetter. But if you learn to use these tools, you can educate yourself in the principles of these fields and you may be able to use that knowledge to your benefit. How many of you would be willing to type a 130,000-word novel (about 600 double-spaced pages) five times over? How about typing it once and then editing and retyping only the parts you feel need changing? More likely the latter than the former. You can do the latter with a computer, but not with a typewriter. Proper application of a computer can lessen the actual time and grunt-work needed to do the job, thus making it more likely you will succeed in completing the project.

The key thing here is that you must initially have the drive and determination to do the work at all.

Could you afford a real typesetting system or a graphics workstation? Would you try to get a job at a magazine or a video-graphics studio just to see if you liked or could do the work? Would you be willing to start at the bottom and maybe someday get the chance to be in a position to

do what you really want to do? Probably not. However, you *could* buy a desktop-publishing program or animation program for your ST and get some experience with the basic principles of the field, and then decide if you want to pursue it any further. You can also do respectable work on an ST for a lot less money than you'd pay for typesetting equipment or a graphics workstation.

This is how personal computers can be used to improve or change our lives. Like anything else worthwhile, it requires effort on our part. The main difference between doing it with the computer or by another means is that the computer may allow us to do it more simply, with less effort and/or less expensively than otherwise, and it allows us to experiment in our own homes.

Paul Dana (author of the *Cyber Star* accessory and an upcoming *CAD-3D*-related product) one day told me that he hated the word "computer" because it doesn't describe what a user does with it. Sure, the computer computes, but when I sit down to animate Megabit Mouse, I don't feel I'm "computing" any more than I'd feel I was "penning" when I'm sketching with a felt-tip pen. Paul said he likes to think of a personal computer as an "enabler" because it enables you to do various things. I agree with him. After all, we don't call a dedicated word processor a "binary data entry/editing machine." A word processor doesn't deal in words; like a computer (which it is), it deals with binary data. It's as much a computer as your ST. But, unlike the programmable ST, it is a computer dedicated to a single task.

Testimonials

A personal computer can be a marvelous thing indeed, if you can figure out not only how to use it, but how to use it to your benefit. Since I've been part of the ST community, I've met, written or spoken to literally hundreds of fellow ST enthusiasts, and I've seen some pretty amazing things.

And, honestly, there's something unique about people who are serious about using their computers. They seem of a different breed than the rest of the population. Computers seem to draw creative and expressive people like a magnet attracts iron filings. I've met writers, editors, marketing people, programmers, professional musicians, artists and educators, a community united under the common umbrella of their interest in

computers (and the ST in particular). Rarely have I seen other fields of interest capable of so passionately uniting such diverse personalities.

Working within the confines of this community, a capable and driven person can make some incredible strides. For example, take my buddy, Andy Eddy... *please!* (Just kidding.) I first started "talking" to Andy on the DELPHI telecommunication system back in early 1987. Andy had been writing reviews for ANALOG Computing, SFLOG (a brand-new magazine then) and also for *Atari Explorer*. He was working for a cable TV company in Connecticut. Within a year of that meeting, he and I had created a pair of *Cyber Studio* design disks, and only a year and a half after that, he'd stepped up to and beyond a monthly column for SFLOG (*Database DELPHI*) to being associate editor of SFLOG and executive editor of VideoGames & Computer Entertainment. Quite a leap, professionally speaking.

As for myself, back in late 1984 I was working for a large mining company, running a computer in a regional field exploration office. That Christmas I bought myself an 800XL. By May I'd purchased a 130XE (then brand new) to replace the 800XL. And then came that fateful day at the tail-end of summer when I plopped down \$937.50 for a just-released 520ST, an SF354 disk drive and SCI224 RGB monitor. About six months after I purchased the ST, I sold my first article to ANALOG ("Pixel Perfect," issue 47). Two months later, I sold another article ("Customizing the GEM Desktop," which later became the fourth *Step 1* article). By mid-'87 I had started writing the first *Step 1* articles. The turning point for me was being able to capitalize on those small successes.

As I was suddenly "press," I was able to make contact with people in the software industry that I'd have had little luck trying to contact had I not been a published writer. One of these contacts, Stephen Friedman, provided me with the then-Beta *Art & Film Director* programs, with which I made the *Star Trek* demo. This led to the *Art & Film* video I made for Epyx. Creating Megabit Mouse for that video got me a lot of work. I managed to capitalize on that, and I've since done game graphics for Omnirent Software, manual designs for CodeHead Software, covers for SFLOG and VG&CE, consulted for Epyx....

Did I mention I quit my job with the mining company in early 1987 and have

been freelancing ever since?

The point here is not to show off or toot my own horn for those of my friends who are in the industry, but to show how people can capitalize on their interest in computers and use that to lever themselves into new experiences, or even new careers.

If you're wondering what the point of this exercise is, this is it: If you use your ST just to play games, it's a game machine. If you only use it for letter writing, it's a word processor. If you only balance your checkbook, it's an expensive pocket calculator on steroids. It will do nothing more for you than you *make* it do. The computer is what you, and you alone make of it.

And that is the final lesson. The first step in doing anything with your computer is to use it.

There is no promise that even the greatest effort in using a computer will ensure your reaching your ultimate goal. There are no guarantees. Then again, through really using your ST you may learn of paths to destinations you'd never before seen or considered. The possibilities are there—if you have the determination to search for and follow them. That's quite a package of possibilities, considering how little you have to pay for it. In my humble opinion, it's more than worth every last penny.

Class dismissed.

The staff of SFLOG would like to thank Maurice for every syllable of those 96,000 words he's written since the beginning of Step 1. He is, without a doubt, the hardest-working author we've ever had the pleasure to work with, and although we wish we could change his mind about not continuing this column, we respect his wanting to get on with other projects. Hopefully, many of those projects will appear in SFLOG. ■



Maurice Molyneux is a longtime user of Atari computers whose interests cover writing, graphics and animation. He enjoys good science fiction, loathes "rap music" (an oxymoron if ever there was one) and loves watching old Warner Bros. cartoons. Ideally, he would like to pursue a career creating animated films.

Experiment 42

or
Edwin: Pre-vivisection

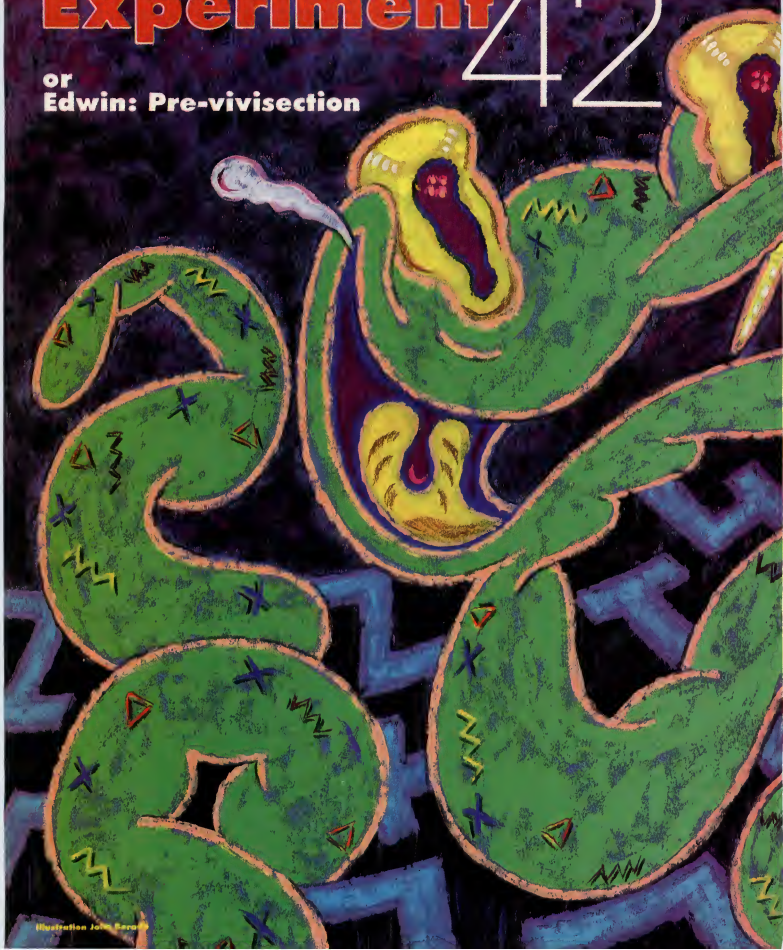


Illustration John Savory



BY GREG KRAUSS
LOW RESOLUTION

"You have the test subject, Wilson?"
"Yes. His name's Edwin."

You hear your name and look around. There doesn't seem to be much different from the last time you looked around, so you stick out your tongue. There doesn't seem to be much different now, either, except that your tongue is sticking out, so you pull it back in.

"Its name is not Edwin. Its name is Biological Intelligence Experiment Subject #42, Lot 6, Reptile Engineering."

"I pity your children. What did you name your daughter? 'Miosis'?"

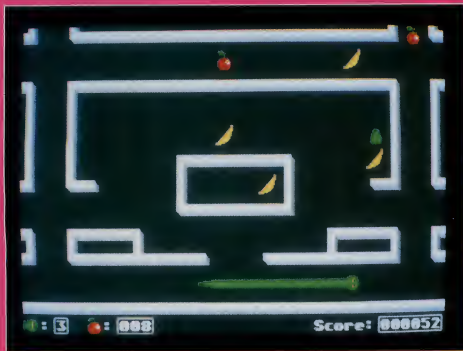
"Funny, Wilson, funny. Just put the test subject in the maze, eh?"

"Yeah, yeah . . ."



"Hellooo,
Edwin."

The only danger in Experiment 42 is dead ends. If you guide Edwin to a spot where his head cannot advance another space, Edwin will bite into his body and lose a life.



Whoa, you think, *something's happening*. You look around in time to realize that you're not sitting on anything anymore. You're rising in the air, and you think it might be wise to stick your tongue out.

"Hellooo, Edwin."

"Wilson! Put it down! Let's get on with it."

"Yeah, yeah. . ."

Now you're going down, and you think it might be wise to pull your tongue back. You are resting on the ground again, but a different ground than you were resting on before. You vaguely think how nice it would be to have hands so you could scratch your head. Or even eyelids to blink quizzically. Instead, you stick your tongue out again.

"This is supposed to be the best the guys in BioEng can come up with? It looks pretty stupid to me."

"Yeah, well, you look pretty stupid to me, so who's to judge?"

"Your professional standards are amazing, Wilson. I am in awe."

"Yeah, yeah, just fill out the form on the

clipboard. I'll start the experiment."

Suddenly, something is very different. Very, very different. Very, very, very different. You decide to stick out your tongue. Something's still very, very, very different. You pull your tongue back. Something continues to be very, very, very different, and now you're sure it's not your tongue. Something red gleams down a passage. *Why, it's food*, you think joyously. *Food! Food! Food! Food! Food!*

"Jeez, look at the way he took off. He must be hungry."

"It's an it, Wilson."

"Whatever."

"I still say it doesn't look very bright."

DeARChing the game

To dearchive (unsqueeze) *Experiment 42*, copy the files EXPER42.ARC and ARCTTP off this month's disk onto a newly formatted floppy or an empty folder on your hard disk. Double-click on ARCTTP and type EXPER42 when the program requests input.

To run the game, double-click on GFABASRO.PRG. When the file selector appears, select EXPER42.GFA.

Playing the game

In *Experiment 42*, you control Edwin the Mutant Snake with a joystick plugged into the ST's joystick port. You navigate him through several laboratory mazes, eating prizes and avoiding dead ends. When you eat a prize—various fruits and other foods—Edwin's body will spend a short time digesting it, then he will grow by one body segment. The more prizes you eat, the longer Edwin grows, up to a set maximum for each level.

When you have eaten the number of prizes required to complete a level (shown on the bottom of the screen by an apple, followed by a number), the level will increase, as will the maximum possible snake length, the number of different prizes that can appear on the screen, and the snake speed. Each maze you complete will also cycle you to the next one until you have finished all five, at which point you will return to the first.

The only dangers in *Experiment 42* are dead ends. If you guide Edwin to a spot where his head cannot advance another space (and that's impossible unless a part of his body is blocking a path), he will bite into his body and lose a life (shown on the bottom of the screen by Edwin's head, followed by the number of lives left.)

If you try to run Edwin straight into a wall, he will automatically turn in an open direction, favoring the left side and top of the screen if he has a choice.

To pause the game during play, press the space bar. The space bar and the joystick button will move you from the title screen into the game, and from the end of the game back to the title screen. To quit a game in progress, press the escape key. Escape will also exit back to the desktop or GFA if you press it at the title page or the final score screen.

The end

Foodfoodfoodfood, you continue to think. Foodfoodfoodfoodsnake.

Oops.

"I told you he didn't look very smart."



Greg Knauss is not 21, not attending the University of California, San Diego, and not telling the truth.

Attention Programmers!

ST-LOG Magazine is interested in programs, articles, and software review submissions dealing with the Atari home computers. If you feel that you can write as well as you can program, then submit those articles and reviews that have been floating around in your head, awaiting publication. This is your opportunity to share your knowledge with the growing family of Atari computer owners.

All submissions for publication, both program listings and text, should be provided in printed and magnetic form. A typed or printed copy of text is mandatory and should be in upper and lower case with double spacing. By submitting articles to **ST-LOG Magazine**, authors acknowledge that such materials, upon acceptance for publication, become the exclusive property of **ST-LOG Magazine**. If not accepted for publication, the articles and/or programs will remain the property of the author. If submissions are to be returned, please supply a self-addressed, stamped envelope. All submissions of any kind must be accompanied by the author's full address and telephone number.

Send your programs and articles to:
ST-LOG Magazine
P.O. Box 1413-M.O.
Manchester, CT 06040-1413

QIAN'S QUEST

BY
NICHAN
CHADWICK

One of the sure paths to success in the software world is to boldly go where everyone has gone before, to paraphrase *Star Trek's* opening line. After one brave soul has broken the ice with some daring effort, many others like to jump in and share the water.

So it's been with *dBASE*, Ashton-Tate's remarkably popular database manager for the PC world (actually, it sprang from that crusty but much-missed dinosaur, CPM). Lots of *dBASE* clones, compatibles and contenders have emerged from the woodwork, few able to garner more than a small share of the market. That hasn't stopped many from trying. Database managers are among the most important programs available, and there's considerable room for variation and improvement.

What made *dBASE* so well received? For one, it broke free from several traditions and gave the user not only an application—the database manager itself—but the tools by which the user could also change the application. *dBASE* is not merely a program, it is a language and an interpreter. You can use it "as-is" or write your own database manager, report printer, field display and so on.

dBASE got an undeserved reputation as a difficult and unfriendly program, a reputation Ashton-Tate worked hard to overcome. They improved the front end by giving it the "Assist" menu/dialog box interface, improved the report and label-creation programs and over the generations added a wealth of commands and functions.

dBASE's language isn't on a par with mega-languages like C; it's rather simple

in comparison. Its focus is data: strings, fields, reports, output. In syntax it's rather like BASIC, and anyone with even a rudimentary experience in BASIC will find it relatively easy to write programs in *dBASE*. Many of the commands are, if not identical, similar. You can enter many of the commands directly or write them into stand-alone programs.

And *dBASE* is quite flexible, tolerant of faults and bug-free. One example is its ability to change the database format after it has been created. Even with a large database stored, it is simple to add or delete fields, change field sizes, modify reports and resort the data—without losing anything or crashing the program.

In response to user comments, *dBASE's* user interface has changed to suit more of the casual or novice users. With pull-down menus and pop-up dialog boxes, it's pretty simple to use. If your tastes run more to the dry functions, simply press Escape and you're in the command-line interpreter. For that matter, there's much more power in the command structure than the user interface allows. It's well worth learning how to use it.

For the ST, the *dBASE* clone is *dBMAN V*, from VersaSoft. It has evolved over many versions, including an early one marketed by Atari itself. The back of the *dBMAN V* manual claims it has "all the power of *dBASE III+* and more on your Atari ST!" That's a mighty impressive claim, one I wouldn't want to defend as literal in court.

Lest you get led astray, *dBMAN* is not 100% *dBASE III+*-compatible. It reads the .DBF database files but has problems with *dBASE* indexes ("Error 39 line 6 program CMDLINE.RUN"), even with SET DB3 ON. It also won't load *dBASE II* report and label files, and several *dBASE* commands are not supported or require a substitute. The user interface is similar to but not the same as *dBASE*.

My first experience with *dBMAN* was less than impressive. I tried to read a *dBASE III* file from drive A, got a disk I/O error and the program crashed. Nastily. From that moment on, it refused to load the "assist" program, the menu-driven user interface, and it crashed back to the desktop every time I tried. I copied a few files from the original disks, but it still crashed. Finally I gave up, erased the lot and copied the original back over again—every single file. Two bombs again the next time I tried. Sigh. I rebooted and tried again. After a lot of disk chunking, it finally worked.

That's not a good way to begin a relationship. *dBMAN* also takes what seems an

endless time to load files and overlays. This is due in part to the glacial speed of the Atari file I/O itself, but I'd hazard a guess that their code could be optimized and some gain in speed achieved. Speed is one of the great bugbears of data processing, and the faster the program works, the happier everyone is. *dBMAN* seems to have a lot of overlays, too. It's 300K-plus but spends a frightening time loading overlays.

If you look in the documentation index for the SET commands, you won't find them. That's odd. There are a lot of them—44 by my count. A little digging through the manual showed me that the nearly 150 functions (as opposed to the commands) are not listed in the index either. Nor are the MODIFY, SHOP, REPLACE, and a whole lot of other commands. Combine this with a lack of a master tutorial and tutorial, and the manual comes across as one of the weakest links in the package. Fortunately, because of its closeness to *dBASE*, a wide variety of third-party books can help provide generic information, although they can't repair the manual's flaws.

Okay, so I finally loaded my *dBASE* file into *dBMAN*. So far, so good. The database appeared to arrive intact (all 40 records). I tried to index the database. The result: "Error 11 line 6 program CMDLINE.RUN. report to VersaSoft." Great. Then it said: "Error 12 line 96 program CRINDEX.RUN." Then "Error 12 line 113 program A.RUN." "Error 11 line 58 program ASSIST.RUN." And instead of returning to Assist, where I'd been when I left, it went back to the command level. So I typed "Do Assist" and got the message "Err 11—A Disk Read Error has Occurred."

Error 11 in the manual says "check if the disk is operational. Check if the diskette is in place." Okay, I quit and checked. Yep. Both okay. Other programs work fine. Back into *dBMAN*. Try to create an index again. Oops. This time I'm told "Sorry, but that filename exists." That's okay; I want to overwrite it (the file has zero bytes). Nope, it won't let me. I have to enter a new filename.

Whatever caused my first problem disappeared, and the index is generated without any error messages. Fine. I close the database and open a larger one: 222 records. It appears to load fine, but when I tell it I want to edit record 222, I'm told "Record 222 is not in the database file!" But I can go to 221 and use Control-N to get there. The record number clearly says 222. When I use Locate and select Bottom, it finds record 222. What's wrong?

Oops. I accidentally selected modify rather than edit. I get a dialog box with "Create oneuse existing one?" There's no cancel option. What if I don't wish to do either? Escape and Undo won't work. I have to wait until I get to the file selector before I can cancel. Not very friendly. A lot of boxes need this sort of addition. And why do some of the pop-up boxes allow mouse selection while others require that I use the keyboard arrows?

Okay, so I want to edit a record. I can enter a record number or accept "0" for the current one. But what is the current record number? It doesn't say. Just have to hope it's the one I want.

Let's create a report. I want to call it "SRT," the same name as my sorted database. When I enter that name, I'm told "File already exists. Use MODIFY REPORT." So I go through the tedious process of getting back to Assist and select Modify Report Form. But there is no "SRTFRM" showing in the file selector! Grrr...

Back to the report generator. All I want to do is create a simple report of the book titles in my database. I enter another report name (why?). A lot of disk chugging again. Sigh. Now I have to enter the master database file. Huh? I wanted to use the one I had open in Assist. And during the report process, I keep having to specify the database name. How many times must I tell it? Every time I insert a field, obviously.

I'm very familiar with *dBASE*; I've used it for a couple of years and written databases, reports, etc., for the likes of the University of Toronto. I can create a *dBASE* report in my sleep—it's that easy. But confronted with *dBMAN*'s report screen, I'm stymied; it isn't an intuitive process. I have to dig into the manual for help and find myself hacking and experimenting because the descriptions and screen shots are not useful or complete.

The report generator is powerful, lots of commands and flexibility, but confusing. There are tutorials for reports and labels, but they're weak; they offer step-by-step help, but not always the reasons for your action. For example, when you create a report or a label, you are asked for a master database file and a master index. In all three tutorials you're told you don't need a master index. Why not? I can't find any other references about using a master index in the manual. But *dBMAN* asks. It's this sort of thing that adds to the aggravation.

The output tutorials discuss only the direct command-line entry method, rather than the menu-oriented system.

Now in my opinion, ST users will prefer to use a menu-based system; otherwise they would have bought a PC in the first place! Since *dBMAN* is available on several platforms, I assume the command-line entry is how they get around having to write multiple manuals. An ST-specific manual is provided, but I can't see that it's very ST-oriented.

In *dBASE* I can reassign the function-key definitions. I don't think I can in *dBMAN*. I can't find any reference to it in the manuals.

When I tried to convert a *dBASE* file, I was thrown back into the GEM desktop with a converted file zero bytes in size.

dBMAN is a diamond in the rough, but it has a lot of problems, mostly in its cumbersome dependence on overlays, the clumsy user interface, the unpolished screen displays, glacial disk I/O and mediocre manual—things that, if cleaned up, would improve the program 1,000%. It's also not bug-free, something I expect from a company that's been slogging away so long at a program.

On the plus side, it is a powerful and flexible database program with a lot of muscle. To get to that core, you need dedication, perseverance and faith. It's certainly the most fully featured database manager available for the ST, and the wealth it offers is undeniable. I just think they could have made it easier to get to it all.

Overall, I'm unsatisfied. The program doesn't feel polished or even entirely finished, not the feel of a thoroughly professional product. Against the competition (*Data Manager ST*, *Superbase* and *Regent Base*), its weaknesses balance the features so that it doesn't offer a significant advantage over the others beyond the *dBASE* file compatibility. I need more *dBASE* compatibility—indexes, forms, labels—before I move my working files from the PC to the ST, not to mention greater speed and a smoother interface. Maybe I'll have to wait for *dBMAN IV*.

By now you probably know about *DynaCADD 1.5*. ISD deciding to release 1.5, a significant upgrade, rather than bring 2.0 out yet. They're concentrating their efforts on producing a multi-platform CADD product, including IBM, Mac and Unix. But *DynaCADD* on the ST needed—and deserved—improvement. So programmer Dave Fletcher spent many hours rewriting code so that 1.5 could come about.

Version 1.5 adds many new features, including Bezier and b-spline curves, hatching, sectioning, a vector font editor (and ten Compugraphic fonts), 2-D solid fills,

major and minor grid and axis marks, named layers, measure area command, filletting between arcs, background plotting, Postscript support and more. They also reduced the program size by using overlays and managed to make loading them fast enough as to be almost unnoticeable. *DynaCADD 1.5* now works comfortably without accessories on a 1040 ST. The new version is also streamlined, with several unnecessary features, such as "Suspend," removed.

A lot has gone into this version, even a major manual overhaul, to the point of eliminating the criticisms I had in my review (co-authored with Thom Weeks). I recommend that anyone who is a serious CADD user and who hasn't picked up on *DynaCADD* yet get this new version. It's a super program, and the only professional CADD package in the market.

A while ago, I received a small book about the ST, called *The Atari ST Book*, by Ralph Turner. Since it is more or less a beginner's book, I didn't pay it too much attention. However, in the past months the book has become a minor celebrity among a small group of novice users here. Unhappy with their official documentation, I recommended they take a look at Ralph's book. Many of them have found it helpful. It's full of useful hints, tips and information the Atari manual either misses or glosses over. There is some peripheral stuff, to be sure, including software-specific discussions that are dated, and I don't agree with his negative view toward protected software; but overall it's a good buy and has a non-technical approach, without talking down to the user. Recommended.

Quick test. What's wrong with this line, found on the back of Michton's *Personal Finance Manager*:

"It's GEM interface allows transactions to be entered or altered as easily as filling out a form."

Got it yet? The possessive of "it" is "its." "It's" is the contraction for "it is." If you failed this test, do your homework. The next test is on the passive voice. ■



Ian Chadwick is a Canadian freelance writer and editor whose hobbies include writing about, reading about and exploring the use and abuse of the English language. He is also an amateur paleontologist, naturalist and carpenter.

The Michigan

Exposition

ATARI COMPUTER

BY DAREK MIHOCKA

By the time you read this, the Michigan Atari Computer Exposition will be several months in the past. But since there were quite a few exhibitors at the show, some of you may be interested in knowing what you missed.

The show was held during the weekend of May 6th and 7th in Romulus, Michigan (about 20 minutes west of Detroit). This location put it within easy reach of most of the Great Lakes states, and this was evident by the large user-group showing from Michigan, Ohio, Pennsylvania and Ontario. Almost a dozen user groups had displays set up. The show itself was presented by the Detroit-based Michigan Atari Computer Enthusiasts (MACE).

In addition to the users' groups, about 25 vendors had almost 5000 square feet of floor space and there were continuous seminars being given on a variety of topics for both the 8-bit and ST computers. During the 15 hours of the show, almost 2,000 people attended.

Being on the heels of the World of Atari Show in Anaheim, not all of the major Atari developers were present.

Most notably, there was no sign of Avant-Garde's *PC Ditta II*, nor of the *Spectre GCR* by Gadgets by Small. And there was no sign of the Transportable or Portfolio machines either.

That is not to say there weren't new hardware products. One that caught my interest was the VID-IST real-time frame grabber from Scotland. Compared with the other video digitizers I've seen, this one is the best—and inexpensive too. It is capable of digitizing three or four images per second in 16 grey levels, and the digitized images are not blurred, even if the objects are moving. The show price was \$14995 (regular price \$19995), although no units were actually on sale there. They are available from Computer Games Plus in California.

For the 8-bit Atari, Chuck Steinman of DataQue was demonstrating the

Turbo-816 upgrade. This board replaces the XL or XE's 6502 chip with the enhanced 65816, a 16-bit chip, giving the computer access to a full 16 megabytes of linear memory and about a 10% increase in execution speed. The board includes a new operating system written for the 16-bit chip, which Chuck claims is compatible with over 90% of 8-bit software. The board was selling for \$110 at the show, while the regular price is about \$130.

DataQue is also working on other products to support this board, including memory upgrades, an 80-column card, a 640x200 graphics card and a 65816 developer's kit. At the time of the show, an Atari 800 version of the Turbo-816 was almost ready. Contrary to some earlier rumors, the board does not drastically increase the speed of the machine. According to Chuck, this is due to speed limitations of some of the chips on the 8-bit motherboard, and a speed increase would require that about 20 chips be replaced.

Datafree Industries was showing off its accelerator board for the ST. The board increases the clock speed of the ST to 16 megahertz, thus potentially doubling the speed of the 68000 chip. In reality, the speed increases are in the neighborhood of about 30% due to the fact that the speed of the ST's memory is not affected. I determined this using the Quick Index software, although I found that the speed increase is not that noticeable by looking at the monitor.

ICD was showing the FAST Tape backup system and, of course, their FAST disk drives. Tom Harker was on hand to give a seminar on the topic of hard-disk drives, which included his disassembling a FAST disk drive and passing it around the audience. Hopefully, he got all the pieces back!

There were plenty of software developers at the show too. Wayne Buckholdt of SofTrek gave a seminar on different techniques to improve the performance of the ST. At the SofTrek booth, he was demonstrating the latest version of *Turbo ST*, as well as a soon-to-be-released product called *Turbo Graphics*, which speeds up all of the ST's graphics functions even more. I conveniently happened to have a copy of my spirograph generator around and tried it with *Turbo Graphics*. Sure enough, it was quite a bit faster. Wayne says that *Turbo Graphics* will be offered as a \$20 upgrade to *Turbo ST* owners.

The Gribnif people had plenty of business at their booth. They were selling the latest version of their *NeoDesk 2.0* pro-

gram, which replaces the normal ST desktop with one that is more powerful and easier to use. The show price was \$35, much less than the \$50 list.

CodeHead software was pushing their products, which now seem to include CodeHead sweatshirts. Their new product, *HotWire*, offers one-touch program execution while coexisting with the normal ST desktop. Charles Johnson (one of the CodeHeads) held a seminar with George Morrison of Alpha Systems, and they discussed such topics as copy protection, piracy and viruses.

Representatives of WordPerfect Corp. were on hand at their booth promoting *WordPerfect*, and George Mella held a one-hour seminar to demonstrate the features of the product. He announced that the price of the famous word processor is being reduced to \$250 (from \$329) due to declining sales in the ST market. He wouldn't give a clear answer as to whether *WordPerfect* will continue to release updates to their ST version, although they will continue to support their current customers.

MichTron had a rather small booth (considering the number of products they have), and George Miller presented a seminar on *Fleet Street Publisher*.

Nice & Software had a booth showing off its *Critic* inventory-control and cash-register software, which comes with an optional bar-code reader. With a GEM-based user interface, the system is an attractive alternative to using a standard cash register.

Atari Corp. was represented, although none of the Sunnyvale people were there. Mike Groh gave an interesting seminar describing the different types of MIDI software available for the ST.

There were many other software vendors selling discounted 8-bit and ST software. They included Joppa Computers, Mars Merchandising, Cal Com and Rite-way. Spectre 128 cartridges, with ROMs, were selling for about \$250. My favorite bargain of the show was the \$15 Atari 1020 color plotter. Remember those? Considering that there is nothing at even ten times the price that lets your 8-bit or ST print text and graphics in color, I just couldn't pass up a bargain.

And finally, some shareware products made their debut at this show. Yours truly demonstrated the latest version of *ST Xformer* (the 8-bit emulator).

Altogether, it was an entertaining show. It certainly allowed a lot of people here in the Northeast to see some new products and meet their developers without having to fly down to Anaheim. ■



Darek Mihocho is a computer-engineering student at the University of Waterloo. His latest ST project is the improved ST Xformer, and he is currently working on image-processing and character-recognition software. His DELPHI user name is DAREKM.

Companies mentioned:

Atari Corp.
1196 Borregas Avenue
Sunnyvale, CA 94086
(408) 745-2000

CodeHead Software
P.O. Box 74090
Los Angeles, CA 90004
(213) 386-5735

DataQue Software
Dept. T-800
P.O. Box 134
Ontario, OH 44862

Gadgets by Small
40 W. Littleton Blvd.
#210-211
Littleton, CO 80120
(303) 791-6098

Gribnif Software
P.O. Box 350
Hadley, MA 01035
(413) 584-7887

ICD, Inc.
1220 Rock Street
Rockford, IL 61101
(815) 968-2229

MichTron, Inc.
576 S. Telegraph
Pontiac, MI 48053
(313) 334-5700

SofTrek
P.O. Box 5257
Winter Park, FL 32793
(407) 657-4611

WordPerfect Corp.
288 West Center Street
Orem, UT 84057
(801) 225-5000

PHOTO: ELLEN SHUSTER



SOFTWARE ENGINEERING

THE SOFTWARE METRIC SYSTEM



How quickly can you write computer programs?
Do you program just as fast in one language as another?
Are you getting faster?
How good are the programs you write?
Is there a way to tell how good is "good"?
Are you getting better?
How large are your programs?
How complex?

BY KARL E. WIEGERS

How quickly can you write computer programs? Do you program just as fast in one language as another? Are you getting faster? How good are the programs you write? Is there a way to tell how good is "good"? Are you getting better? How large are your programs? How complex?

If you have answers for all (or even any) of these questions, you should be writing this series, not reading it. These questions all pertain to measurement: size, speed, quality. These are characteristics of software production, as they are characteristics of any creative or manufacturing process. However, people are still grappling with good ways to "measure" software. As an introduction to the topic of software quality-assurance, let's spend a few pages thinking about software-measurement techniques, also known as "software metrics."

Why measure?

Recall from earlier installments in this series that the major challenge facing us is the ability to produce high-quality software rapidly enough to meet an ever-increasing demand. The key concepts are quality, reliability and productivity. We've talked about methodologies for improving all of these factors, including structured analysis and system specification, structured system design, structured programming, structured testing and computer-aided software engineering (CASE).

But a critical question remains: How can I tell if the quality and productivity of my software development efforts are improving? This is where the metrics come in. We need some way to assess software quality, such as the number of errors per quantity of code. Productivity usually reflects the quantity of a product created per unit effort. Reliability is indicated by the number and severity of failures per unit time. This still leaves some questions, such as how to measure the quantity of software created in a project, what constitutes an error, and how to quantify the effort that goes into a software project.

There are several benefits to being able to measure our software production in these ways. A chronic problem is that software systems are delivered to the customer much later than expected. Good project management demands reliable methods for estimating the size, complexity and work effort needed for a software system. We can only base such estimates on our previous experience, so collecting software metrics can help us better estimate the completion time, cost and resources required for the next project. This is extremely important for business planning.

From a technological viewpoint, it's important to know which of the various methodologies aimed at improving productivity are providing the best payback. If the CASE tool you're using doesn't seem to be increasing your productivity as much as you'd like, you might want to spend your money on some other kind of programming assistance. This assessment is based squarely upon meaningful methods for measuring productivity. Software metrics can also help assess the magnitude of our maintenance burden by tracking the cost, size and impact of changes in existing systems.

The problem

It's much more difficult to measure the manufacture of software than it is for hardware (by which I mean almost

anything besides computer programs). Let's use the term "widget" to refer to any kind of non-software product. What are some differences between widgets and software?

A widget is some kind of object, but what is a "software," or even a "program"? You can't count the software equivalent of widgets directly. Also, it's difficult to apply to software the statistical quality-control measurements people do with widgets. Defects in widgets are usually attributable to flaws in materials or production. There are ways to spot the defective widgets before they are sold and to decide how to fix a poor process. But all copies of a computer program are identical (unless there's a hardware flaw, like a bad floppy disk). The defects in software are intrinsic in the product, and they're usually hard to detect.

Given all these problems, let's see some methods people have devised for measuring different aspects of the software development process.

How much?

Lines of Code. We'll begin by discussing how to measure the size of a software system. The number of lines of code (LOC) has long been used for this purpose. LOC has the advantage of being an objective measure that is fairly easy to determine, but there are several shortcomings.

One question is simply the definition of a "line of code." Should you count comment lines in the source file? What if you have several logical statements on the same physical line in the file (a poor programming style, but you may remember it from older BASICs)? How should you deal with continuation statements, permitted by some languages, in which one logical line is split over several physical source lines? Should statements like variable type declarations in C or Fortran count the same as actual logic or computation statements? What about lines of text in a help display?

Perhaps the simplest measure for LOC is to count each nonblank, non-comment logical source statement in the file as a single line of code. Continued statements count as one LOC, no matter how many physical lines they occupy in the source file.

For a given programming language, using lines of code to compare the relative sizes of program modules is usually okay. However, if one module is longer than another, you can't tell whether the longer one is more complex than the other or just less efficiently coded.

We run into more problems when we try to compare modules written in different languages. An assembly language program generally takes several times as many source statements to accomplish a task as does a higher-level language, which is why we use the latter whenever we can. To be sure, the assembly statements are usually much shorter, but we haven't considered the complexity of each line of code yet. In fact, this is another shortcoming of the LOC metric: All source statements are not created equal.

Some new languages don't even involve lines of code in the conventional sense. Fourth-generation languages (4GLs) often involve some kind of query language by which users can access databases without having to write all their own routines for handling files, presenting information on the screen, performing calculations, and so on. If you develop an application using a 4GL, you may end up with few actual "lines of code" in a system that performs some complex tasks. It's not meaningful to compare LOC in a case like this with a system written using a 3GL, like C, Pascal or BASIC. Obviously, LOC as a software size metric has some limitations.

Number of Modules. Another way to estimate the size of a software system is by counting the number of modules it contains. Recall that we define a module as a named, callable block of code. Again, however, this metric doesn't consider the complexity of different modules in a system, or of modules in different systems. Nor does it consider pre-existing modules that were reused in this application.

The modules I write in high-level languages average about 30-50 lines of code (according to the definition I gave a few paragraphs back). In contrast, I recently heard about a huge software system written in PL/I, totalling about one million lines of code and about 1,000 modules. This works out to an average of around one thousand LOC/module. (Naturally, I have no idea how those folks defined "line of code.") Simply counting modules in these two cases isn't a good comparator of system size.

But to a first approximation, especially if the same software developer is involved in each case, we can compare the size of software systems by counting the number of modules, number of reused modules, the average lines of code per module, the number of lines of help text and the number of elemental pieces of data involved in the system. This last notion gets back to the system data dictionary.

ary we've discussed in earlier articles. The October 1988 issue of *STLOG* might be worth reviewing if you're fuzzy on data dictionaries.

Function Points. Some years ago, A. J. Albrecht at IBM came up with another scheme for estimating the size and complexity of a software system, that attempts to circumvent the language dependency. This method is based on the fact that every program performs an assortment of specific functions. Some of these will be simple, and some more complex. Albrecht's method counts the "function points" contained in a software system. Albrecht's original papers would be hard to find, but you can read about the function-point method in Roger Pressman's book, *Software Engineering, A Practitioner's Approach*, 2nd Ed., McGraw-Hill, 1987 (pp. 91-94). Since the function-point metric seems to have broad applicability, let's talk about the method a little bit.

To tally the function points in your software system, you go through four steps. First, count the number of instances of five different components: external (user) inputs; external outputs; external inquiries; logical internal files; and external interface files. Table 1 defines these five component types. Next, classify every instance in each of these classes as to its relative complexity: low, average or high. Albrecht's papers contain guidelines for the complexity classification based on the number of individual data elements and the number of file types referenced in each component. We won't worry about the details now.

The third step involves weighting the instances of each component type for complexity. Table 2 shows the weighting factors used. Let's suppose your system contains five external inputs of low complexity, three of average and one of high complexity. Applying the weighting factors from the first row of Table 2 gives this equation:

$$(5*3) + (3*4) + (1*6) = 33$$

This means that your system contains 33 function points from external input-type components. Perform the same calculation for the other four component types, applying the appropriate weighting factors from Table 2, to come up with the "unadjusted function points" for your system.

The final step is to compute a fudge factor, which you'll apply to the unadjusted function-point count to determine the final number of function points. This adjustment number is based on 14 factors, listed in Table 3. You as-

sign a numeric value from 0 (no effect in your system) through 5 (strong effect in your system) for each of these 14 factors. Then, add up these numeric values for all 14 factors, divide by 100 and add the result to 0.65. This is your weighting factor, which will range from 0.65 through 1.35. An "average" system is imagined to have a weighting factor of 1.00, corresponding to an average value of 2.5 for each of these 14 items. In other words, our fudge factor can change the raw function-point tally by plus or minus 35%.

Multiply this weighting factor times the unadjusted function-point count, and there you have the final, adjusted function-point count for the system. This equation below summarizes the computation process:

$$FPs = \text{raw FP sum} * \\ [(0.65 + (0.01 * \text{fudge factor sum}))]$$

Whew! I agree that the function-point calculation is tedious. But in practice it works out well. After you've gone through the computation once, you'll get the hang of it. Function points do provide a measure of system size and complexity that is independent of language. In fact, you can even estimate the function points for a system at the design stage, which is helpful for estimating completion times for the project if you know how long it takes you to generate one function point worth of code. Which brings us to the next class of software metrics.

How fast?

Now we get to the question of how to measure productivity. We want to be able to measure it because we have a stated goal of improving our productivity. Unless we know where we are, we really can't tell if we're making any progress.

"Productivity" can be defined as the ratio of product created to the effort expended in creating it. In the preceding section we looked at ways to estimate the quantity of software created for a particular project. Despite their limitations, we'll have to use one or another of those metrics for the numerator in any productivity calculation.

One obvious measure of effort expended is the time spent on a software project. This is surprisingly difficult to measure accurately. First, of course, we have to define the start and finish points. The software engineering philosophy suggests that the official beginning of a project might be the time at which you sit down to write the statement of purpose for the system you are about to build.

The ending point is more obscure. An arbitrary definition is the time at which the product (documentation and all) is delivered to the customer. This doesn't mean that no more work will be done on the project after delivery. Rather, this definition simply identifies a boundary between the system-development process and the infinitely long maintenance phase. You're probably interested in measuring both development work effort and maintenance work effort, so it makes sense to draw a line between the two.

Another consideration is how to count hours during the development period spent on activities other than working directly on the project. We can think of two kinds of development time. "Gross" time includes all work hours (weeks, months, years) between the beginning and end of the project, including meetings, coffee breaks, vacations and chitchat. "Net" time counts just the hours devoted explicitly to project activities. The difference between gross and net time is the necessary overhead for having human beings perform a task. The employer has to pay for gross time, but he might be able to charge the customer only for net time. Because of such nightmares, I've avoided the business aspects of software development as much as possible.

Let's assume we can measure the time spent on a project to our satisfaction. Now we can talk about productivity. A simple measure is lines of code written per day. Before you get too excited about this metric, remember the shortcomings of the LOC measure itself. You can expect to see big differences in LOC/day for programs written in different languages.

A longtime software industry benchmark is that an average computer programmer can generate only 10-15 lines of debugged code not per hour, but per day. This seems abominably low, but by the time you factor in the time spent on system analysis and design, progress reviews, testing and documentation, it turns out to be sadly accurate.

Alternatively, we could calculate productivity by considering function points to be the measure of product created. Therefore, function points created per unit time (day, week, month) is a productivity metric. We would expect this metric to depend less on the language used than does the LOC/time metric.

The opposite of productivity is cost. If you imagine cost to be proportional to time (as in hourly salaries for program-

mers), we can get cost metrics by taking the reciprocals of our productivity metrics. These might be hours/line of code or days/function point, which could be translated into \$/FP or \$/LOC if you have a cost/hour figure available.

You may think this notion of cost doesn't apply if you're a computer hobbyist working on your own, rather than a cog in a corporate wheel. But it does! Your spare time isn't really "free" time; it's valuable to you. What if you're spending an hour a day at the Atari keyboard when you have the option of working overtime at your real job? Don't tell me there's no cost associated with that.

What do you do once you've selected a productivity metric? The first step is to apply it to your current or most recent projects. The idea is to build a baseline of your current productivity status. Then track the productivity metrics for your future projects and compare them to the baseline. If you see your productivity increasing or your cost decreasing, congratulations; you're becoming a more efficient software developer.

A decrease in productivity could mean lots of things, so analyze it before you panic. Are you confident that your baseline measure is accurate? Are you working in a new environment (unfamiliar computer or new language)? If you're in a team environment, are there new members on the team who aren't fully up to speed yet? Are the metrics you're using the most appropriate ones for the work you do and the factors you care about?

You can use the productivity measure to determine how changes you've made in your development effort are working out. For example, I recently worked on a project where we used a new CASE tool for system design and a new programming language on the IBM PC. We also experimented with another tool intended for increased productivity: a code generator. After six man-months of effort, I calculated our productivity at, you guessed it, ten lines of code per day! What a disappointment.

But the project wasn't a failure. The learning curve associated with the new environment is bound to cost some productivity. For the CASE tools, I viewed this as an investment that will pay off in spades on future projects, when the learning curve is gone but the benefit remains. More important, I felt that our software engineering approach had resulted in a great improvement in the quality and reliability of the system we created. So despite the apparent lack

of productivity gain, I concluded that this project, and the application of the new SE technologies, was a grand success.

For personal programming of the sort most hobbyists do, productivity takes on a slightly different meaning. You probably don't care so much that every hour is spent as efficiently as possible, because you're doing it all for fun. But the notion of productivity can become important if you're trying to justify making further investments in your hobby. Remember, your personal time is worth money. How much time would it save you to have a hard disk or a second floppy drive? How about a more powerful C compiler? Of course, all these arguments fly out the window if you just want to buy a new toy (nothing wrong with that), but they can improve your chances of success when attempting to convince a spouse that you need to spend a couple of hundred bucks.

In the professional world, productivity metrics are important for estimating completion time and the cost of a new project. Suppose that, based on a system specification and preliminary design, you estimate that a new program would comprise 60 function points. You know from collecting metrics that you can generate an average of three function points per week. Thus, you estimate that this project would take about 20 man-weeks to complete. Now you have some meaningful numbers to present to your customer. We'll talk more about software project planning in a future article.

How well?

Another class of metrics addresses the question of software quality: How many defects are contained in our products, when are they discovered, at what point in the development cycle are they introduced, how costly are they to correct and what is their impact on the system? Lots of questions here, but the answers are important to software quality-assurance. If you know where your software errors are arising, you can concentrate your efforts in the right places to minimize the defects.

Software defects

What do we mean by a "software defect"? Basically, I'm referring to some unanticipated and undesired behavior in the system. (Occasionally you'll encounter some unanticipated but desirable behavior, called an "undocumented feature," but those are scarier than hen's teeth.) You know these better as "bugs."

It turns out that most software defects

are errors of omission, not commission. By an error of commission, I mean something like a syntax error or an erroneous algorithm. Modern compilers prevent most syntax errors, such as incorrect function or subroutine argument lists, from slipping through. They do have some trouble reading your mind, however, so mistakes in algorithms probably won't be detected. An error of omission means that you've left something out, like a trap for bad input data, or a check for a full disk before attempting to write to it, or an ELSE statement in an IF/THEN construct. This sort of mistake is best avoided by following the software-engineering stratagems of systematic design, review and testing. The software-engineering approach preaches a twofold assault on bugs. Bug prevention is best accomplished by following structured analysis, design and implementation techniques. Early bug detection is facilitated by software quality-assurance efforts, including structured walkthroughs, project reviews and testing. Software quality-assurance will be the subject of a future article.

It's useful to keep track of when in the development cycle defects are identified and at what phase they were introduced. It's much easier and cheaper to correct errors when they are detected early in the development life cycle. As time goes on, undetected errors become better concealed in the thicket of code that grows over the skeleton of design. Also, the range of influence of a particular bug widens as its tendrils penetrate into more parts of the system. The longer this infiltration continues, the more difficult it is to surgically excise the critter without killing the patient.

Defect metrics

Let's assume that you've devised a method for counting defects that show up in your systems either in testing or in the hands of users after the system has been released (horrors!). One way to quantify the defect rate is to count defects revealed per line of code. I hope that you can do better than one error per line, so a more useful measure is defects per thousand lines of code (defects/KLOC). Track this metric over several projects; if you see a decline in the defects/KLOC figure, pat yourself on the back for having attained a quality improvement.

But wait! We're only counting the defects we've spotted. How many more are lurking about that we have not yet encountered? This is impossible to know with certainty, but it's a sure bet that a

module that has been found to contain a large number of errors probably contains even more than you think. You should monitor the defects/KLOC/unit time, since bugs are gradually revealed with continued use of the program. As the maintenance phase continues, the rate of appearance of new bugs normally dwindles.

Measures used to assess the severity of errors in software (and hardware, for that matter) are the mean time to failure (MTTF), mean time between failures (MTBF) and mean time to repair (MTTR). MTTF is the average time that the system runs properly before it crashes due to a defect. This may be indicative of the density of bugs in the system. The MTTR pertains to the difficulty of fixing a bug once it is detected, sort of a measure of how difficult maintenance on the system is. The mean time between failures is the sum of the MTTF and the MTTR. These metrics can be used to estimate software-system reliability.

How complex?

The final category for software metrics today is module complexity. We've already seen that lines of code and func-

tion points tell us something about system size, but not much about the complexity of a particular module. One method was devised by Thomas McCabe. McCabe's complexity measure is based on the notion of representing the control structure of a module (branching and iteration constructs) in the form of a graph and then counting particular features of the resulting graph for each module. We won't worry about the details now.

Another computer scientist, M. Halstead, devised equations for quantitatively calculating the complexity of a program module by counting operators and operands. Operators are things like equals signs, comparisons ($<$, $=$), IF statements and mathematical symbols ($+$, $-$, \times , $/$). Operands are variable names or constants used in comparisons or loops. Both the numbers of unique operators and operands, and the total number of operators and operands that appear in the module are used in the complexity calculation.

I've never actually used either McCabe's or Halstead's complexity metrics, so I won't go into more detail. The important point is to know that methods for calculating module complexity do exist, should you ever need to do so.

The bottom line

We've talked about several metrics that attempt to quantify different aspects of software creation: quantity, productivity, quality and complexity. If you do try to use some of these metrics, it's important to get your definitions straight at the outset. At least this way you can be internally consistent, either for your personal projects or for systems built by all the people in the same organization. It's somewhat dangerous to use these met-

rics to compare different software developers, since the numbers are so fuzzy and since they ignore other important aspects of software engineering, such as documentation and coding style.

Nonetheless, I feel that anything we can do to quantify the software-creation process will give us a possible handle for improving that process. Software metrics are an important aspect of any serious software-engineering effort. ■



After receiving a Ph.D. in organic chemistry, Karl E. Wiegers decided it was more fun to practice programming without a license. He is now a software engineer in the Eastman Kodak Photography Research Laboratories. He lives in Rochester, New York, with his wife, Chris, and two cats.

Table 3

Adjustment factors for function-point analysis.

1. Does the system involve data transmitted over communication facilities?
2. Does the system perform processing on more than one computer?
3. Is system performance (speed) a critical feature?
4. Does the system run in an existing, heavily used computing environment?
5. Is the system designed to handle a high transaction rate?
6. Does the user enter data online (as opposed to in batch)?
7. Does the design of an online system emphasize end-user efficiency?
8. Are the logical internal files updated by online activities?
9. Does the system involve particularly complex processing (mathematical computations, heavy error-checking)?
10. Is the code in this application designed to be reusable?
11. Is the system designed for easy conversion to production and easy installation?
12. Does the system require reliable backup and recovery operations?
13. Was the system designed to be used in multiple installations by multiple organizations?
14. Was the system designed specifically to facilitate change in data or files by the user?

Table 1.

System component types for function-point analysis.

External Inputs: Count each input by which the user supplies application-oriented data to the system. Each data input screen counts as a single input, even if it has multiple data elements.

External Outputs: Count each output from the system that provides application-oriented information to the user. This could be screen displays, printed reports or error messages.

External Inquiries: Count each kind of request the user can make for the system to do something, such as retrieve and display data, mouse-clicks, showing help displays, etc. Not the same as external inputs.

Logical Internal Files: Count each logical grouping of data used within the system; similar to data stores.

External Interface Files: Count each file that is shared with another application, rather than being strictly internal to the present system.

Table 2.

Complexity weighting factors for function-point analysis.

Component Type	Low	Average	High
External Inputs	3	4	6
External Outputs	4	5	7
External Inquiries	3	4	6
Logical Internal Files	7	10	15
External Interface Files	5	7	10



BY ARTHUR LEYENBERGER

I still have not gotten over the new, new Atari. First there was the "old Atari," then after Jack Tramiel took over, the "new Atari" arrived. After a couple of years it seemed like the same "old Atari" all over again. Now, as promised by Atari themselves, 1989 has become the year of the new "new Atari" (it gets confusing, doesn't it?).

After years of Atari's neglecting the U.S. market, ignoring users and generally having problems from DRAM chip supplies to product announcements, they seem to be getting their act together. I won't rehash all the problems and the apparent fixes to these problems; they have all been discussed here many times before. I'll just say, on behalf of the ST user base, we're glad Atari is better focused than they ever have been before. These are exciting times for Atari users once again.

You probably know about the ST Laptop, now called the Transportable. (The name was changed from Stacy.) The Laptop product has given Atari the potential to finally enter the mainstream computer market. With its appeal to MIDI users, the capability to run Mac software with the

Spectrum 128, PC compatibility with PC Ditto and portability, the Laptop is an exciting product that should attract many users.

Once Atari enters the mainstream market, it will be subject to the needs of these types of users and the peculiarities of the market. Atari will have to do more than have a mainstream product—they will have to become a mainstream company. It will be a challenge—and one that will require many changes in the way the company does business. I hope they are up to it.

The "Are you kidding?" department

Through the years, I have discussed the growing trend of software and hardware manufacturers relying on the courts rather than the market to promote, protect and sell their products. You know what I am talking about: Apple Computers threatened lawsuit against Digital Research (DR), which caused DR to redesign the GEM Desktop (on PCs); Lotus Developments sued Paperback Software for supposedly infringing on the look and feel of 1-2-3; etc.

It's one thing to legitimately protect the copyrights and patents that a company holds against imitators and thieves. I support the companies that do this because it is rightful and correct. But it is another thing altogether to stifle what could have become a standard in the interest of greed. This type of behavior I detest, and have said so many times.

The latest round of this silliness comes from Tandy. Early this year, the company acquired Grid Systems, an early producer of MS-DOS laptop computers. Last spring, the Grid subsidiary of Tandy filed suit against Toshiba for infringement. Grid is seeking three times the amount of lost business and asking the court to prohibit Toshiba from selling laptop computers that infringe on its patents.

At the heart of the issue are three patents granted in 1985 and 1986 that Grid holds for the hinge design and case of portable computers. Toshiba, Zenith, NEC and a dozen other laptop companies have been making computers with a screen that folds over the keyboard when not in use. This "clamshell" design has become an industry standard for laptop computers in the last few years.

The laptop computer market is expected to surpass \$400 million in sales in 1989, and Toshiba is the largest supplier in the market. That is why Grid chose them to test their case. If they are successful, Grid could collect millions of dollars

in damages and future royalties. It's interesting to note that Tandy had several laptop products of their own prior to the purchase of Grid this year. It would be somewhat amusing if the Grid subsidiary decided to sue their parent company.

Will Atari be affected? That is unclear. The ST Transportable uses the familiar clamshell design and could conceivably be a target of Grid, once it gets finished with the major players. In any event, the case could take years to settle if it reaches the courts.

Intersect news

By the time you read this, Intersect Software will have started shipping *Masterlink*, their "next generation" telecommunications program. *Masterlink* has been totally rewritten to incorporate many new features, such as background operations, script language, remote operation and much more. Because of its added features and power, a minimum configuration of one megabyte of memory is required.

Masterlink has an easy-to-use script language that can be edited using either its own buffer windows or your favorite word processor. The script language offers dialog boxes, conditional branching, string manipulation and much more. Scripts allow you to automate everything from dialing an information service and signing on to downloading or uploading files. A script language and a good telecommunications package offer the user a lot of flexibility and power.

Another feature of *Masterlink* is its ability to run another program while you are, say, downloading a file in the background. Unlike *Interlink*, which requires a well-behaved GEM program to operate properly, *Masterlink* can run even a TOS program simultaneously. You can even be running a program while someone else is accessing your ST via the remote BBS option.

Masterlink now has more loadable Transfer buttons and a Batch button in the File Transfer Window. Other protocols can use the Batch button as well. The File Transfer window offers ten transfer protocols, of which seven can be loaded from separate transfer protocol files. All file transfers take advantage of the Background Manager, and the complete status of any transfer can be seen from this window, which is accessible from within any program.

The Style/Features window offers five loadable emulation protocols via separate buttons on the screen. Each button can be preloaded with your favorite terminal

emulation from a script, the Startup Parameters Window or by double-clicking on the one you want from a list. There is also a Noise Filter button at the bottom of the window to provide cleaner communication.

Other features of *Masterlink* include 40 dialer buttons, access to just about every GEM menu function through icons, three editor windows (with word wrap, tabs and reformat), a fast find/replace function and the ability to cut and paste between the buffer windows. As you can gather, *Masterlink* is a very complete telecommunication package for the ST.

Masterlink is not intended to replace *Interlink*, which Intersect will still be selling and supporting for owners of 520 STs. *Interlink* continues to be improved, and the latest version (1.85) offers a faster method of loading protocol files. Instead of using the Recorder, the new version has a change to the Setup window that allows emulation and transfer protocols to be brought in when the program is loaded.

Version 1.85 of *Interlink* also has a fix for problems encountered when accessing CompuServe at 1,200 baud. It seems that CompuServe has been changing their 1,200 nodes all over the country, which prevents *Interlink* from automatically dialing at 1,200 baud. If you do have an older version of *Interlink* (before 1.85), you can use the Recorder to dial and log onto a CIS 1,200-baud node, as long as you don't use the dialer to connect yourself to CIS.

To upgrade to the latest version of *Interlink*, send your original disk and \$5 to cover shipping and handling to: Intersect Software, 2828 Clark Rd. Suite 10, Sarasota, Florida 34231. *Interlink* users can also upgrade to *Masterlink*. Instead of the normal \$60 list price, registered users can upgrade to *Masterlink* for \$23.

Interlink also has a new version of *Revolver*, which is a program that allows you to swap several programs in and out of memory via user-controlled partitions. Version 1.1 is the latest edition of *Revolver* and offers improvements to existing features (some are bug fixes) as well as a new boot manager that allows you to choose which programs (auto and desk accessory) you want resident when the current partition boots up.

Features of Version 1.1 include a fix for the GEM 40-folder problem (the original version of GEM can have a maximum of 40 folders), ability to turn off the *Revolver* mouse accelerator, smaller minimum block size for partitions, addition of a partition copy command and the ability to show folders when using a wild-card pat-

tern for a file name. In addition, the new version forces a floppy-disk media change whenever you change partitions, adds a folder rename command to the list of disk commands, allows you to exit *Revolver* dialogs with the Return key and forces partitions less than 160K of memory to become TOS-only partitions. Users of older versions of *Revolver* can upgrade to Version 1.1 by sending their original disk and \$6.25 to Intersect.

Hard-disk package

Michton, a longtime producer of ST application and utility software, has just announced their *Hard Drive Turbo Kit*. Selling for \$60, *Turbo Kit* contains three hard-disk products in one: back-up, optimize and disk cache. The kit contains updated versions of what were originally three separate programs: *Backup*, *Tuneup* and *M-Cache*.

Hard disks are extremely useful peripherals for the Atari ST. However, since they can contain ten to 60 megabytes of information, a problem with the disk can become catastrophic. The question is not if the hard disk will fail, but when it will fail.

Michton's *Backup* utility allows you to backup the programs and data on your hard disk to ensure against future problems. There are two parts to the program: a backup portion that allows you to save the hard-disk files to floppy disks and a restore program that will copy the files from the floppies back onto the hard disk. Several options are provided to give you maximum flexibility.

Backup can perform either a file or an image backup of the hard disk. A file backup is a file-by-file copying of the files from the hard disk to the floppies. File-oriented backups can be done in either DOS or TAR format.

TAR format is faster, but disks written in this format can only be used by the backup system. DOS format disks can be used directly with any GEMDOS program. An image backup is an exact copy (sector by sector) of the hard disk, and can also be used only by the backup system.

Backup lets you perform a full backup of the entire hard disk, an incremental backup of only those files that are either new or have been changed since the last backup, or a date backup. A backup by date will copy only those files that have been created or modified after a user-supplied date. In addition, all backup types (except the image backup) produce a "history report" that contains information about the backup process and a list of files included in that specific backup

and where they are located on the floppy set.

Other options include the ability to format disks during the backup process, determine whether hidden files should be included in the backup and whether data written to floppies should be verified by the operating system for correctness. Also, file-segmenting can be turned on or off. When allowed, a file from the source disk may be spread across as many as four floppy disks.

Tuneup lets you optimize your hard disk for faster performance. During the normal process of adding, deleting and modifying files on your hard disk, the files themselves become fragmented; that is, the contents of the file itself are no longer stored in a contiguous block but in several blocks on the hard disk. This slows down access of the files because the hard disk must skip to each section when it reads these files.

Tuneup does two things to the hard disk. First, it rewrites the files to contiguous blocks on the hard disk, which significantly speeds up the process of reading files. Second, the utility moves all of the hard disk's free space to one contiguous section as well. You can choose to optimize the hard disk for either read access or read/write access.

The third part of the *Hard Drive Turbo Kit* is a disk-cache utility called *M-Cache*. Still another way to increase the performance of your hard disk, a disk cache holds recently accessed disk sectors in memory, much like a RAM disk. This increases the speed of access to that information because it is faster to read from memory than it is to repeatedly read from the hard disk itself.

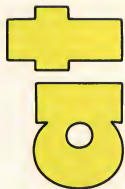
If you use a hard disk with your ST, the *Hard Drive Turbo Kit* can significantly improve its performance and usability. It can be ordered directly from Michton, 576 S. Telegraph, Pontiac, Michigan 48053, or purchased from your local Atari ST software retailer. For information about this and other Michton products, contact them at (313) 334-5700. ■



Arthur Leyenberger is a longtime Atari user and freelance writer living in beautiful New Jersey. He can be reached on CompuServe at 71266,46 or on DELPHI as ARTL.

FEATURE

ATARI



The curious, the adventurous and the enthusiastic all appeared at the Disneyland Hotel last April for two days of displays, conferences and entertainment. Three aisles of manufacturers and retailers filled the hotel's convention facilities. At times, walking from exhibit to exhibit was difficult due to the more than 7,000 attendees who squeezed into the hall.

JIM HOTZ AND MICK FLEETWOOD

Disneyland

THE ANAHEIM!

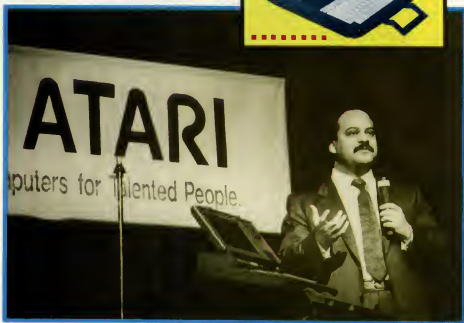
ATARIFEST!

by Frank Cohen



Because Atari ST sales have lagged behind those of the Amiga personal computer, the World of Atari show was an important event for Atari Corp. At first, Atari would not commit to attending the show, but later they rented two booths and brought much of its California sales force to Los Angeles. Placed in a corner of the hall, its back against the wall, Atari turned the show to its favor by displaying a fresh new mix of products and technology.

The Transportable (previously named Stacy), the new ST laptop computer from Atari, made its first public appearance. One prototype unit was shown (at arm's length) running a simple slide-show program. The Transportable is a battery-powered 1040 ST computer with monochrome display, and includes a trackball, which substitutes for the mouse, on the right of the full-size QWERTY keyboard. The supertwist LCD display folds down over the keyboard for portability. The unit weighs less than 15 lbs. with batteries installed, and one charge lasts over five hours with medium disk use. Joe Mendolia, Atari's vice president of Marketing, promised the enthusiastic crowd that the Transportable will begin shipping by the end of the summer.



Atari unveiled Word Flair (\$149.95), an integrated word processing, database and graphics program developed by Blue Chip International. The program uses Atari's Graphics Device Operating System (GDOS) to display and print stylized fonts and graphics. Laser-printed output is excellent using GDOS; however, business users working without an Atari SLM804 laser printer will probably be discouraged because of lengthy GDOS printing times on dot-matrix printers.



- **SAM TRAMIEL**
- **TALKS ABOUT**
- **THE TRANSPORTABLE**
- **(A.K.A. STACY)**

ATARI at Disneyland



**MICHAEL PINDER OF THE MOODY BLUES
WITH ATARI'S FRANK FOSTER**



ARTISTECH—DA VINCI

Other vendors

FAST Technologies introduced Turbo16, a 16-Megahertz (Mhz) accelerator upgrade for ST computers. The upgrade package replaces the ST's 68000 Central Processing Unit (CPU) with a faster version of the same chip, which can run software up to two times faster with little or no side effects. Desktop-publishing and CAD users waiting for screen redraws will rejoice when viewing Turbo16's impressive performance.

Installation is fairly easy, although a qualified service technician is recommended. The package is approximately the same size as a normal 68000 CPU chip, but slightly taller, allowing room for some support chips, which patch the ST operating system to eliminate problems found using experimental upgrades. For example, disk functions work as usual. The ST cartridge port and Mega ST expansion cards are also not affected. Even sound generation is maintained with the upgrade functioning.

Special high-speed cache memory has been designed into Turbo 16, which further improves application speed. The extra memory uses low-power static memory that does not tax the ST power supply. At a list price of \$399, Turbo16 is not inexpensive; however, power ST users will find Turbo16 a necessity.

Wuztek demonstrated Omnimon Rainbow, a multi-resolution color monitor. A mode button located on Omnimon's front panel switches the ST computer among low, medium- and high-resolution modes. ST users who commonly switch between color and monochrome monitors will find Omnimon a welcome hardware accessory. Omnimon's screen uses a dark tint, nonglare surface with unusually clear colors.

Best Electronics demonstrated a new ST/Mega-compatible mouse, the CBM1 (\$4995 List), which has an ergonomic design that fits smoothly into your hand. The mouse is easier to control and provides greater movement resolution for finely detailed mouse control. Special Teflon rollers on the steel photo-optic interrupter shafts give the mouse a smooth feel and reduce periodic cleanings. It is even FCC certified as a Class B computing device.

THE ANAHEIM ATARIFEST!

Best Electronics also introduced an upgrade kit for the standard Atari ST mouse that replaces the photo-optic sensors that communicate the mouse direction to the ST computer. Teflon rollers improve the mouse action and reduce cleaning frequency. The upgrade kit has a suggested retail of just \$1595.

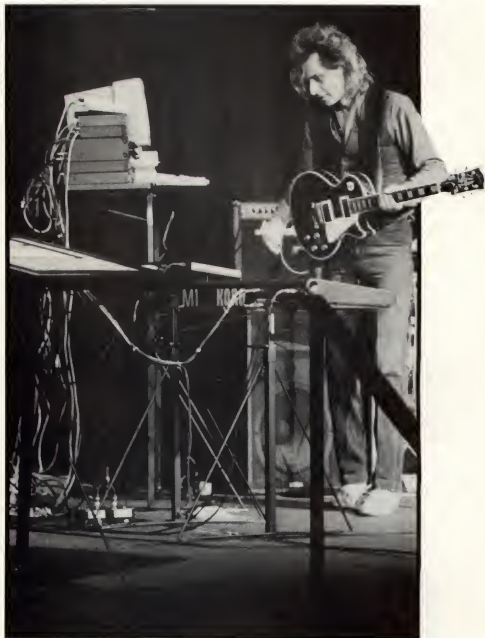
Hard drives on the move

Atari announced a 44-megabyte hard-disk drive that features a removable disk platter. These platters allow your data to be moved from drive to drive, and security-conscious users can keep sensitive data with them when the computer is not in use. The Atari removable plugs into the DMA/hard-disk interface and may be daisy-chained with other hard disks. The device is manufactured by Sybold, a large producer of hard-disk equipment, and is installed into a modified Atari Megaflo 30/60 cabinet. First shipments of the Atari removable are expected later this year.

Westco Electronics is already selling a removable-platter hard-disk drive. The Infinite Storage System (ISS) products offer high-performance, high-reliability removable cartridge-based hard-disk drive units for the ST. The drives access data in 28 milliseconds, about average for normal hard drives. The disk drive is built around ISS Bernoulli Technology, a proven system that is virtually head-crash-free and very rugged. In case of problems, Westco offers a one-year warranty on disk cartridges and a two-year warranty on ISS-2 drives. Atari usually offers only a 90-day warranty.

The ISS drives are compatible with all ST, Mega ST, Spectre 128/Magic Sac and PC Ditto storage formats. Cartridges are disk-format compatible with PC DOS cartridge disks, allowing easy data interchange between the ST and IBM PC compatibles. There are 21-, 44- and 45-megabyte models available, and prices range from \$1,200 to \$1,385.

ICD demonstrated the new FAST Tape Backup system. The new system saves the contents of a hard drive onto a special audiocassette. Backup speeds can be as fast as 6.5 megabytes per minute, a considerable time savings over floppy-disk backups. The tapes are specially coated with a protective layer of acrylic so they run



THE MIDI CONCERT

ATARI at Disneyland

faster and withstand high temperatures. They also cost \$34.99 each.

FAST Tape Backup units have the same footprint as the Megafile 30, and the unit plugs directly into an ST's Hard Disk/DMA port. DMA Out allows the Backup unit to be daisy-chained to other devices, such as hard disks, laser printers, etc. ICD also sells hard-disk-equipped units. A battery-powered clock is built into the controller board. Also, individual file recovery is possible using the GEM-based backup software.

Not to miss the backup boat, SeymourRadix introduced DVT (\$24995 List) a videotape backup system for hard-disk users. DVT works like the ICD FAST Tape backup; however, backup information is

stored on videotapes. DVT stores data at eight megabits per minute, 20% faster than audiotapes.

DVT plugs into the ST as a standard cartridge, and RCA cables attach to your videotape recorder (VTR). DVT software moves data from your hard disk to the videotape by recording data images. DVT restores backed-up data by decoding the recorded images. The system stores up to 360 megabytes of data.

RONSAT Technologies demonstrated *Stonehenge* (\$23995), an externally powered RAM disk. *Stonehenge* plugs into the ST DMA/hard-disk drive port. To the user, *Stonehenge* appears as a disk-drive device on the GEM Desktop; however, since RAM memory is used, *Stonehenge* stores or retrieves files up to ten times faster than the quickest hard drive. Since the unit has a separate power supply, turning off your ST computer will not cause the loss of data, a problem common to RAM disks.

Software developers will find that *Stonehenge*'s speed and after-crash data integrity make it an attractive place to keep development tools, source code and other important data. There are 256K and two-megabyte versions of *Stonehenge* available.

DAVID SMALL OF GADGETS BY SMALL



Spectre tessellates

On the software end of the show, enthusiasts crowded around the first public demonstration of *Spectre GCR*, an upgrade to Spectre 128, a cartridge-based Macintosh emulation system for the ST.

Group Coded Recording (GCR) is the method used by Apple computers to record data onto a floppy diskette. Atari equipment uses the IBM disk format, which is incompatible with GCR. Previous attempts to read Macintosh disks from Atari ST drives resulted in marginal performance. Although *Spectre GCR* failed to read several Macintosh diskettes provided by show attendees, its inventor claims benchmark tests show *Spectre GCR* reading disks faster than normal ST format disks. *Spectre GCR* is undergoing Federal Communications Commission (FCC) testing, and once approved, will carry a \$29995 suggested retail price.

Spectre means Macintosh compatibility. So some Atari manufacturers are beginning to distribute traditional Macintosh products. Sotae Corp., a company that began by selling Macintosh OS

THE ANAHEIM ATARIFEST!

ROM chips for Spectre and Magic Sac customers, sells the Abaton fax modem for the Macintosh. The Abaton InterFAX 1248 fax modem (\$495.00 List) permits the ST to send and receive fax messages. A special software package schedules unattended transmissions any time of the day or night, logs fax activity, transmits documents to multiple locations and maintains a customized fax directory with up to 800 FAX numbers.

The fax modem uses modular telephone jacks to plug directly into any standard phone connector. It is CCITT Group 3 compatible, meaning it can communicate with most modern fax machines, and transmits at 4800 bits per second (BPS). The modem may also be used as a standard 1200/300 BPS modem to communicate with local bulletin-board systems, CompuServe, GEnie or other online information services.

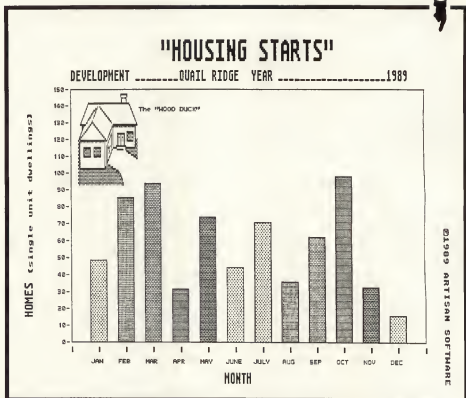
ArtisTech Development displayed *Da Vinci* (\$99.95), a sophisticated graphic drawing program that uses an intuitive menu system displaying a choice of hundreds of pen styles, shapes, patterns, functions and tools. Color or monochrome bit images are edited in several magnification modes without restricting the available tools. *DEGAS*, *Neochrome* and Amiga IFF picture files may be loaded and saved.

Da Vinci simulates an artist's drawing easel with drawing modes that give the artistic feel of a true painting medium. For example, pen tools allow drawings to be smeared with existing images. Other tools include blend, tint, smooth, cycle and filter. Animation sequences are easily created with Delta Compression techniques used in *Aegis Animator* and *Cyber Paint*. A public-domain stand-alone animation player is included.

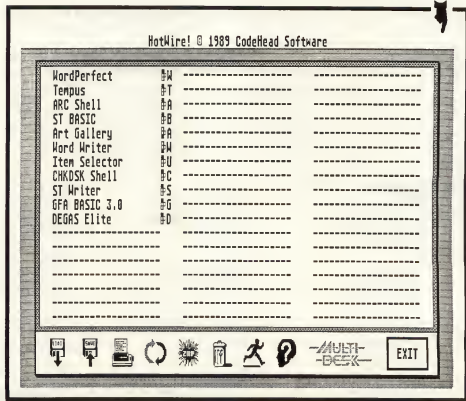
Spritstein (\$299.5 List) is a new game-creator utility for GFA BASIC. The utility helps would-be game designers create screens, maps and sprites, the necessary parts of every video game. Everything is mouse-driven and little knowledge of games programming is needed. The package includes fully commented source code compatible with GFA BASIC (Versions 2 and 3).

An optional data disk for *Spritstein* contains several game maps, sprite definitions and commented source code.

GRAPHMAKER



HOTWIRE



Sprite Disk Volume 1 (\$24.95 List) demonstrates *Spritstein's* ability to create arcade and adventure-style video games.

Artisan Software demonstrated *Graph Maker* (\$59.95 List), a GEM-based color business graphing program. *Graph Maker* receives lists of numbers, summarizes the results and displays (or prints) bar charts, line graphs and pie charts. Graphs may be saved in *DEGAS* or *Neochrome* format for later editing. Desktop-publishing users should have fun with this one. Clip art may also be used in *Graph Maker* drawings.

Shell game

NeoDesk 2.0 (\$49.95 List) is Grinbif Software's solution to many GEM Desktop shortcomings. For example, where are the keyboard equivalents for GEM Desktop functions? *NeoDesk* replaces the GEM Desktop with a graphic shell that looks and feels like the normal GEM Desktop. But *NeoDesk* goes further, adding a wish list of functions that should have been written into the GEM Desktop.

Many *NeoDesk* functions are based on the Macintosh desktop: customized icons for each program or file type, files are moved when dragged, rather than copied, scrolling functions auto-repeat when the mouse button is held down and applications outside a file's folder may be installed. *NeoDesk* includes a new Control Panel desk accessory with a screen-save utility, visible clock display, blitter-skip control and free-memory display. The screen-saver blanks your ST display after a user-defined number of minutes pass without user activity. Move the mouse or click a key, and the screen returns to view.

Previous versions of *NeoDesk* were reported to use much of the ST computer's memory. The new system runs as

a shell; when an application is opened, *NeoDesk* releases memory, leaving less than 24K devoted to itself. *NeoDesk 2.0* supports a new icon editor used to create unique icons for each file or program. The icon editor is GEM-based and easy to use.

The shell game has raised the rhetoric level among GEM Desktop, *NeoDesk* and *HotWire* enthusiasts. Each camp claims the others are poor excuses of true user friendliness.

Charles Johnson and John Eidsvoog (CodeHead Software) demonstrated *HotWire* (\$39.95 List), a keyboard macro system for opening applications. Press a user-defined "hot" key from the GEM Desktop and an application is automatically opened. And to think that using the mouse to point-and-click was once considered revolutionary.

HotWire finds applications hidden in folders, no matter where they reside, with the *HotWire* menu displaying up to 54 programs at once and showing hot-key icons and 20-character application descriptions. The program, written completely in fast, compact assembly language, uses 40K of your ST's memory.

CodeHead also demonstrated *MultiDesk* (\$29.95 List), a desk accessory handler; *G+Plus* (\$34.95 List), a GDOS replacement; and *MIDI-MAX* (\$49.95 List), a real-time MIDI performance tool.

Ditto ditto (or Son of Ditto)

Avant-Garde Systems answered customer complaints of slow IBM emulation speed by introducing *PC Ditto II* (\$299.95 List). When *PC Ditto* was first introduced, much praise was garnered because it apparently worked as promised, running all MS-DOS software. However, the emulator

worked 30-50% slower than the slowest IBM PC computer. Avant-Garde's new solution comes as an inexpensive hardware-based emulation system that solves the slow time factor.

PC Ditto II features up to 640K usable memory, monochrome and color graphics capabilities on all Atari monitors, fixed disk adapter with automatic access to all Atari partitions, Microsoft-compatible mouse, Atari clock/calendar data and time support, full serial- and parallel-port emulation, and support of 3 1/4-inch, 80-track drives and optional 5 1/4-inch 40-track drives.

Uncle Atari

The air is buzzing once more with the word "Atari!" The ST has been called a NeXT killer, Macintosh crusher and PC destroyer. In truth, the ST computer has become the computer industry's favorite uncle, most likely not invited to a company board meeting but always around at family get-togethers. On arrival from out of town he is welcomed with open arms. Such was the case at the World of Atari, the latest of the new Atarifest conventions.



Frank Cohen is a programmer, author, graphic designer and music hobbyist. You may contact him directly on CompuServe (76004,1573) and GEnie (FRANKCOHEN), or by writing to PO Box 14628, Long Beach, CA 90803-1208.

Products mentioned in this article:

De Vinc
ArtisTech Development
P.O. Box 214830
Sacramento, CA 95821
(916) 488-6844

DVT
Seymour/Radix
P.O. Box 166055
Irving, TX 75016
(214) 823-5797

FAST Tape Backup
ICD
1220 Rock Street
Rockford, IL 61101
(815) 968-2228

Graph Maker
Artisan Software
P.O. Box 849
Manteca, CA 95336

HotWire
CodeHead Software
P.O. Box 74090
Los Angeles, CA 90004
(213) 386-5735

Interfax
Sotae Corp.
6150 Jessup Road
Cincinnati, OH 45247-5842
(513) 385-3312

I55 Removable Hard Disk
Westco Electronics, Inc.
4695 S. 1900 W #6
Ray, UT 84067
(801) 773-8447

Mause Plus
Best Electronics
2021 The Alameda, Suite 290
San Jose, CA 95126
(408) 243-6950

NeoDesk 2.0
Grinbif Technology
P.O. Box 350
Hodley, MA 01035
(413) 584-7887

Omnimon Rainbow
Wuztek
4521 Campus Drive, Suite 400
Irvine, CA 92715
(714) 733-0189

PC Ditto II
Avant-Garde Systems
381 Pabst Point Drive
Jacksonville, FL 32225
(904) 221-2904

The Transportable
Atari Corp.
1196 Borregas Avenue
Sunnyvale, CA 94086
(408) 745-2000

Turbo16
FAST Technology
14 Loveloy Road
Andover, MA 01810
(508) 475-3180

Spectre GCR
Gadgets by Smail
40 W. Littleton Blvd. #210-211
Littleton, CO 80120
(303) 791-6098

Spritstein
ASTRA Systems
2500 S. Fairview, Unit L
Santa Ana, CA 92704
(714) 549-2141

Stonehenge
Ransel Technologies
368 Lexington Drive
Buffalo Grove, IL 60089
(312) 520-8003

► NOTE: Due to its large size, this program is available only on this month's disk or in the database of the STLOG ST users' group on DELPHI.

The Compukid Connection: Dot-to-Dots

LOW RESOLUTION



PHOTO: LOUIS CASTENEDA

by D. A. Brumleve

Dot-2-Dots is a computerized version of the familiar dot-to-dot puzzle game. Designed for young children in the pre-school and early primary grades, this version allows the child to manipulate the characters in the dots so that a sequence of numbers, upper-case or lower-case letters may be followed when solving each puzzle.

Successful completion of a puzzle is rewarded: The child is offered a color palette from which to choose colors to paint the on-screen puzzle picture. If a dot-matrix printer is available, the child may print out a puzzle for solving on paper or for coloring and embellishing with crayons. Dot-to-dot puzzles can be great entertainment for young children,

and the extra features of *Dot-2-Dots* enhance that enjoyment.

In addition to the dot-to-dot game for the very young, a puzzle editor is provided for use by an adult or an older child in creating additional puzzles. (Puzzle creation can be fun, too!) Both the game and editor programs will run in low-resolution on any color ST.

Getting started

To use the programs, first unarchive the DOT2DOTS.ARC file on your SFLog disk (instructions for this are provided on the disk in the DEARC.DOC file) and copy DOT2DOTS.PRG, DOTMAKER.PRG and DOT2DOTS.DAT to a freshly formatted disk. The files DOT2DOTS.LST and DOTMAKER.LST contain the GFA BASIC listings of the programs; they are needed only if you want to examine the source code.

Your disk should remain in the same drive at all times during play so that new puzzles can be loaded. The programs will operate from a folder, provided all three files are present in that folder. The programs will also operate from a hard drive.

Be sure to

SHOW your puzzles before saving them.

This often reveals deficiencies in a design that will

not be noticed by looking at

the design grid.

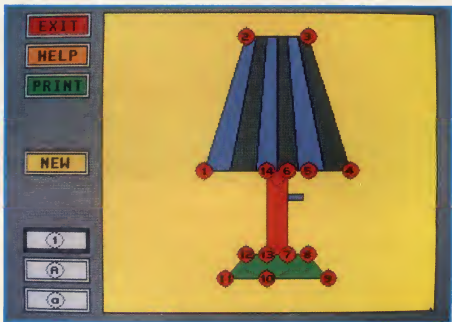
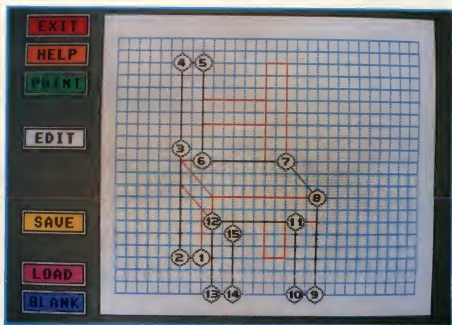
The game

Double-click on DOT2DOTS.PRG. After a brief wait, you will be presented with a title screen. Click CONTINUE and a dot-to-dot puzzle will be loaded automatically. A mouse arrow on the screen moves to each consecutive dot in the puzzle, demonstrating how to play the game. When the puzzle is complete, the arrow fills the puzzle box with color.

Following this on-screen tutorial, the main screen appears and a puzzle is automatically loaded for play. To play, simply click on the white dots in numerical order. If you click on the wrong dot or on no dot at all, you'll hear a low tone. If you click on the expected dot, the dot turns red. Upon successful completion of the puzzle, you are presented with an opportunity to fill the puzzle with color. When you are finished coloring, click DONE to return to the main screen and begin a new puzzle.

The main screen has seven options: EXIT allows you to leave the program gracefully; HELP provides a review of these main screen options.

PRINT allows you to print a puzzle in one of two ways. The "Dots Only" option will print unconnected numbered or let-



tered dots, while the "Lines Only" option prints the lines connecting the puzzle dots, but not the dots themselves. In either printing mode, any extra lines provided by the puzzlemaker (not necessarily connected to the dot points) will also be printed. Children enjoy printing out puzzles to be colored by hand with crayons. If you have trouble with your printout, please refer to the "Printer Problems" sidebar. To load a new puzzle, click on NEW. The computer selects a puzzle at random and displays it.

The last three white option boxes allow you to change the dot numbering/lettering system. For example, if you'd like to work a puzzle with uppercase letters inside the dots, just click the "A" icon. Upper- or lowercase letters, or the numbers 1-26, can be used. The letters are mostly in the standard Atari font, but some were changed because they were difficult for small children to identify. (Over 200 lines of source code are devoted to the redesign of these letters.)

All options are available at all times (except when you are coloring the puzzle). For example, if you begin work on a puzzle and decide not to finish it, just click on NEW to load a new puzzle. If you decide halfway through that you would rather work with the letters "a" through "z" than with numbers, just click on the "a" icon.

The editor

The key to enjoying Dot-2-Dots is, of course, in the quality of the puzzles; uninteresting puzzles will not challenge a child for long. Because the reward for successfully completing a puzzle is the fun of filling it with color afterward, puzzles should be divided into several fillable compartments whenever possible.

Puzzles must also be geared to the age and skills of the child. A child who can count no further than ten will be frustrated by a puzzle of 26 dots. It is a good idea to arrange the puzzle dots from left to right (or clockwise, if appropriate), providing important eye exercise for pre-readers. The interests and experiences of the child can suggest the subject matter; a child who loves dinosaurs will also love your brontosaurus puzzle.

When completed, the puzzles should resemble something, not just be a mess of points and lines, because part of the fun of working with dot-to-dot puzzles is in guessing what each puzzle represents. A variety of puzzles is provided in DOT2DOTS.DAT on your STLOG disk, but after several plays, a bright child will be able to guess what each puzzle is on

sight. New puzzles will be needed to maintain his/her interest. For this reason, DOT MAKER.PR3 is offered to enable you to create additional puzzles for the amusement of your young one. The user-interface of this editor is suitable for use by children as young as eight, but offers enough features to satisfy the requirements of adult puzzlemakers as well.

The main screen of DOTMAKER.PR3 is very similar to that of the game itself. Instead of a blank white puzzle area, there is a blue design grid to help with the alignment of dots and lines. The EXIT, HELP and PRINT options function the same as in DOT2DOTS.PR3. In the game program, all the options are available at all times. In the editor, however, certain options will not function unless you have a puzzle in progress. For example, the PRINT option is not available when there are no puzzle dots to print; clicking on PRINT when the design grid is blank will result in no reaction from the computer.

SAVE allows you to add a puzzle to the DOT2DOTS.DAT file on your disk. When you click SAVE, you'll be asked whether or not you want the first and last dots of the puzzle to be connected by a line. A further alert will confirm that you want to save the puzzle and give you a chance to change your mind. Each puzzle adds 255 bytes to DOT2DOTS.DAT.

LOAD will load an existing puzzle from the puzzle file. If you click LOAD repeatedly, you can see all the puzzles in the file in the order in which they were saved. BLANK erases whatever puzzle is currently on the design grid. No alert boxes will caution you when you click LOAD or BLANK; I found that too many alerts made working with the program annoyingly slow. Instead, these options were placed at the bottom of the screen, where they are unlikely to be selected accidentally.

The actual editor screen, from which puzzles can be created or altered, is reached by clicking the EDIT option on the main screen. When the editing menu appears, the DOTs option is already selected. Your arrow cursor changes to a crosshair, which has a hole in the center so you can see the color of the pixel below it. If you click on the blue grid area, a puzzle dot will appear under your cursor. Clicking a second time places a second dot, and a line will connect the two dots.

Now choose the LINE option. This option is used to add extra lines to your puzzles—lines that will not be drawn automatically as puzzle dots are connected. Click on the grid and a tiny red dot

will appear. This is the beginning point of a line. Click again to choose the end point of the line, and a red line will connect the points.

If you now click UNDO, the entire line will be erased. UNDO will erase the last drawing action unless you have changed from the DOT mode to the LINE mode (or vice versa), or have changed a dot using ALTER, or have returned to the main screen since that action. Clicking on SHOW does not interfere with the functioning of UNDO. If you click UNDO after drawing a puzzle dot, the dot and any connecting line will be erased. If you click on UNDO after drawing the beginning point of a line, only that point is erased.

Puzzles often need a bit of fine-tuning. If you want to adjust the position of a dot or remove it completely, click on ALTER. You will then be prompted to identify the dot to be changed. After clicking on the dot, choose "Replace" or "Remove." If you choose to replace the dot, you will be prompted to click on a new place on the grid for repositioning. (If you decide that you prefer the original position of the dot after all, simply "Replace" it in the same spot.)

The grid and the colored lines serve a useful purpose, but they do not show you exactly how the puzzle will look when loaded into DOT2DOTS.PR3 for play. In addition, the first and last dots are not connected on the design grid, but you may save your puzzle so that a line will appear between these two dots when the player chooses the final puzzle dot in the game program. To see the puzzle as it will look to the player, click SHOW. You may examine the puzzle with dots and added lines (these are the red ones you add while in LINE mode); this is the way the puzzle will be presented when the player clicks NEW in DOT2DOTS.PR3. You can also view the puzzle with the lines connecting the dot points, but with no dots. That is the way the puzzle will look when the player is rewarded with a chance to color it in DOT2DOTS.PR3.

There are some limitations on puzzle design. One of the most obvious is that only 26 dots can be used. This number was chosen because there are 26 letters in the alphabet. I considered for a time attempting to put double letters ("AA" or "aa") inside the dots, but the size of the screen would limit the number of dots anyway, and I have found that 26 dots are sufficient for most puzzles. When more points are needed, they can be obtained by adding extra lines. An example is provided in DOT2DOTS.DAT. I didn't have enough dots to make the teddy bear puzzle.

zle as I wanted it, so I added arms in the LINE mode.

There is also a limit on the number of lines—only 50 lines can be drawn (defining 100 points). I have found this to be generous; only in the apple tree puzzle did I use all 50 lines. There is also a limitation on the placement of dots so that they are not too close together. If you try to place a dot too close to an existing dot, the program will alert you. Children need large targets for the mouse, and the dots are relatively small. In the dot-to-dot game program, children are permitted to "miss" the dot by a pixel or two, so there must be some space allotted around each dot to allow for this. There is no limitation on the placement of extra lines. Dots and lines can be placed on the outer edges of the blue design grid, but not on the white area surrounding it.

Be sure to SHOW your puzzles before saving them. This often reveals deficiencies in a design that will not be noticed by looking at the design grid. When I prepared the chair puzzle, for example, I omitted one of the rear legs. If I hadn't viewed the picture with the "Lines Only" option, I wouldn't have noticed the error until the chair puzzle was used in a game. I've also found that it is a good idea to draw the extra lines after you are certain you have the dots exactly where you want them. I use two disks when I design. One is a temporary disk on which I save each puzzle with only dots—before I have added any lines. That way, if I'm dissatisfied with the placement of lines, I can always load the dots-only version back into the program and try to place the lines again. Don't be disappointed if your first few attempts look silly; puzzle creation takes one's thinking, and a bit of practice will help tremendously. Your child will appreciate your efforts! ■



The mother of five children, ages 2 to 10, D.A. Brumleve is involved with children and computers in a variety of ways. An avid programmer, she has developed a beginner's course in GFA BASIC and is the author of PreSchool KidProgs (MichTron) and numerous freely distributed programs for young ST users.

PRINTER PROBLEMS

Both the editor and the game program use a command for a screen dump in order to print a puzzle. This procedure will provide an excellent copy of a puzzle from most dot-matrix printers. To test your printer's ability to accept a screen dump, first make sure your printer is on and connected (and loaded with paper!). Then, from your ST's desktop, press and hold the Alternate key and at the same time tap the Help key. (Just tap the Help key; if you hold it down, the printing process will keep cycling on and off.) Your printer should kick on and print out a picture of the screen.

If the printout shows the entire screen, including the right edge, printouts can be made with DOT2DOTS.PRG and DOTMAKER.PRG without difficulty. If, however, the test printout omits the right edge of the screen, you will need to do the following:

- * Copy the Control Panel accessory (CONTROL.ACC) that came with your ST onto your DOTS2DOT disk. The DOT2DOTS disk will become a boot disk for use whenever you work with either of the two programs. (If your computer boots from a drive other than A, copy CONTROL.ACC to your boot disk.)

- * Reboot your computer with the disk in Drive A.

- * When the desktop appears, pull down the Desk menu and click on "Install Printer."

- * Change the "Pixels/Line" setting from the default of 1280 to 960.

- * Save your desktop.

Whenever you use DOT2DOTS.PRG or DOTMAKER.PRG, reboot your computer from the boot disk. Your printer will automatically be set up to print a full-view screen dump. You will not need to change the "Pixels/Line" setting again, as that has been saved in a DESKTOP.INF file on your disk.

A common problem with printouts involves the left-to-right adjustment of the puzzle. It is my intention that the puzzle box be centered on the printed page. The program is designed to make this the most likely position, but if your printout is off-center, you may want to adjust the position of your paper before printing.

THE

YOU NEED
DISK!ONLY
\$9.95
EACH!

If you want to get the most out of ST-LOG, you're also going to want to get your copy of the disk. Each issue's disk contains all the exciting programs for that issue, including the programs whose listings could not be included due to space considerations. The ST-LOG disk version is truly an excellent software value. Order yours today!

SEPTEMBER
1989

Experiment 42,
Magniwriter ST,
RWABS Monitor
and more!

JULY
1989

MicroCheck ST,
Game
Cupboard,
Animated GFA
Input and more!

MAY
1989

Line Attack,
Outline Plus,
Monochrome
Gray II and
more!

AUGUST
1989

ST Bingo,
Calorie Counter,
Menus with
Pascal and
More!

JUNE
1989

Ballbuster,
MIDI Capture
Safe Keeping
and more!

APRIL
1989

Multi-Point,
Font ID Editor,
Monochrome
Gray and more!

YES

I DO WANT
THE
DISK!

ONLY \$9.95 EACH

- ☐ ST-LOG April 1989 DISK
☐ ST-LOG May 1989 DISK
☐ ST-LOG June 1989 DISK
☐ ST-LOG July 1989 DISK
☐ ST-LOG August 1989 DISK
☐ ST-LOG September 1989 DISK

Name _____
 Address _____
 City _____ State _____ Zip _____
☐ Payment Enclosed—Charge My ☐ VISA ☐ MC
 # _____ Exp. _____
 Signature _____



Add \$1.50 postage and handling for each disk ordered.
 Make check payable to: LFP, Inc. P.O. Box 87000, Los Angeles, CA 90087
 California residents add 8.5%

RWABS Monitor

DAVID SMALL AND DAN MOORE

As you may know, RWABS is a favorite hangout of Dan's and mine. Some people go to bars, some to nightclubs, but Dan and I hang out at RWABS, drinking Boodles gin and Stros beer, hacking the night away.

Ah, the romantic life of a computer programmer.

What's RWABS? RWABS is short for "read/write absolute." It's the place the system goes for any disk I/O. It's also the place people like Dan and I hook into to do things involving the disk.

RWABS Monitor is a utility program that monitors RWABS. Any time RWABS is called (meaning any time a disk request occurs), *RWABS Monitor* puts a small message in the upper right-hand corner of your screen, telling you about the request.

Typing it in

Listing 1 is the ST BASIC program that will generate your copy of *RWABS Monitor*. The resulting file, *RWABSMON.PRG*, is not a BASIC program; however, it is instead 100% machine-language. Type in Listing 1 using *ST Check* (found elsewhere in this issue) to check your work, and run it with ST BASIC. The file *RWABSMON.PRG* will be written to the disk in Drive A.

The complete assembly language source code was too large to print in this magazine. It is available on this month's disk version or from the databases of the ST LOG ST users' group on DELPHI.

Using the program

At first, this program seems sort of like "gee whiz, that's nice, but . . ." Then you begin to see its usefulness.

For instance, when your hard disk begins the fill-up, slow-down problem, you'll first see it on *RWABS*. Disk errors also show up in *RWABS*. It brings you up to date on what your disk is doing. Better, since *RWABS Monitor* is always active, it's active in the middle of other programs, so you can see what they're doing as well. It's like having a ringside seat at DOS working with the disk.

RWABS Monitor also demonstrates some interesting programming techniques. To make it work, I had to do some novel programming—stuff you can use in your code:

1) **Trap #1** interceptor: This code looks for all *RWABS* calls and intercepts them, triggering the screen display if it's an *RWABS*. Ever wanted to know how to write a trap interceptor? They're mighty useful. An easy adaptation of this code would let you direct a copy of all console output to the printer, like for SID debugging printouts. Nice, eh?

2) **Direct screen I/O** code: This neat piece of code lets you output strings, bytes, words, longs in hex and other things to a monochrome monitor or color mo-

onitor directly. This is extremely valuable during debugging. Sometimes you need a way to output (let's say) a variable without using the operating system. This happens *all the time*. For instance, you can't output when GEM is busy, when the disk is busy, when you're in a time-limited interrupt, when a graphics screen is up, and on and on.

Using the font driver included, you can output characters and data anywhere you like at any time. A little work will adapt it to any C or other application. This code works by using the ROM tables that contain character images and copying them selectively to the screen. The code to do color screen updates is particularly hairy because of how the color screen is laid out. It wasn't a lot of fun.

Originally, the code was done for the "crash" display on the Magic Sac, but I've improved it a lot since then. Since there is no trace of the Atari operating system left when the Sac is running, the display code had to completely stand alone. (Still, if you want to see what a chunk of the Sac looks like, here it is.)

3) **Terminate and Stay Resident**: how to keep your program hanging around after it ties itself into the operating system so the next program to be run doesn't get rid of it.

Using the program

To use *RWABS Monitor*, either double-click on it or put it in your AUTO folder. When run, it'll give you a sign-on display. Then it goes to work. With the next disk access it plots a status line at the top right of your screen. You get:

1. Drive letter doing the access (A-P).
2. Which sector the disk access starts at, in hex.
3. How many sectors are being requested, in hex.
4. Where the data is headed for in memory, in hex.

It won't work in low-resolution color because there isn't enough room to pack all

that information in.

It's fun to watch *RWABS Monitor* work, particularly on a hard disk or RAM disk. Accesses fly by. Try doing a show-info, for instance; you'll see all kinds of activity.

Let's take one sample (and common) problem and diagnose it with *RWABS Monitor*.

Your disk is getting a "Disk A is getting an error, Cancel/Retry" message. You want to find out what's wrong, so you load *RWABS Monitor* and go try out Disk A. You see that Sector 1 is being read and read. There's a long pause. Then you get the disk-error message. Conclusion: Sector 1 is zapped. Go to a disk-fix program and correct it.

Or, in a program load, you see Sector 45 being reread many times just before a TOS ERROR 35 message. Again, Sector 45 is your problem; that portion of your program has died. (A good check-disk program can also help in this diagnosis.)

Finally, if you have a hard disk that has more than a few megabytes of data on it, you'll see the write slow-down problem in detail. TOS will painfully and slowly read the FAT sectors, looking for an empty place to put the data. When it finds an empty area, it'll write the data extremely fast. (Of course, if you're using a program like *FATSPED.PRG*, you don't have this problem.)

See what I mean about a front-row seat at disk access?

Conclusion

It may seem esoteric. It might seem like useless information. But after a time, *RWABS Monitor* grows on you. It really helps you keep an eye on your disks.

My luck with disks of all sorts has never been good, so I feel anything I can do to monitor disk drives is good. They're generally the flakiest part of a computer system. *RWABS Monitor* has a permanent home in my AUTO folder. I hope it finds a home in yours.

RWABS Monitor Listing 1: ST BASIC

```

100 OPEN "R", #1, "A:RWABSMON.PRGM", 16: FIE
101 L6 AS BS
110 AS: "": FOR I=1 TO 16: READ US: IF US=
"X" THEN 140
120 A=VAL("R&H"+US): PRINT "W": AS=AS+CH
R$CA: NEXT
130 LSET BS=AS: R=1: PUT 1, R: GOTO 110
140 CLOSE I: PRINT: PRINT "ALL DONE!"
1000 data 60, 1A, 00, 00, 00, 2C, 00, 00, 00, 0
0, 00, 00, 00, 00, 00
1010 data 00, 00, 00, 00, 00, 00, 00, 00, 00, 0
0, 00, 00, 61, 00, 05, EC
1020 data 45, F9, 00, 00, 00, 00, 00, 00, 00, 0
0, 00, 00, 0A, E4, 33, FC
1030 data 00, 00, 00, 00, 00, 00, 0A, E6, 61, 00, 06, F
8, 45, F9, 00, 00, 01, 00
1040 data 61, 00, 06, EE, 26, 2F, 00, 04, 61, 0
0, 07, 24, 45, F9, 00, 00
1050 data 01, 00, 00, 00, 00, 06, DC, 20, 6F, 00, 0
4, 20, 08, 06, A8, 61, 0C
1060 data 06, 8C, 00, 00, 00, 01, 00, 61, 00, 07, 0
6, 45, F9, 00, 00, 01, A2
1070 data 61, 00, 06, BE, 3F, 3C, 00, 7F, 4E, 4
0, DF, FC, 00, 00, 00, 02
1080 data 91, FC, 00, 00, 00, 02, 23, C8, 00, 0
0, 00, 9C, 48, 79, 00, 00
1090 data 02, 08, 3F, 3C, 00, 26, 4E, 4E, DF, F
C, 00, 00, 00, 06, 4E, 71
1100 data 20, 6F, 00, 04, 20, 28, 00, 0C, 00, 0
C, 00, 00, 01, 00, 42, A7
1110 data 2F, 00, 3F, 3C, 00, 31, 4E, 41, 00, 0
0, 00, 00, 00, 00, 00, 00
1120 data 00, 00, 00, 00, 00, 00, 0A, 2A, 2A, 2A, 2
A, 2A, 2A, 2A, 2A, 2A, 2A
1130 data 2A, 2A, 2A, 2A, 2A, 2A, 2A, 2A, 2A, 2
A, 2A, 2A, 2A, 2A, 2A, 2A
1140 data 2A, 2A, 2A, 2A, 2A, 2A, 2A, 2A, 2A, 2
A, 2A, 2A, 2A, 2A, 2A, 2A
1150 data 2A, 00, 0A, 2A, 20, 20, 20, 20, 20, 2
0, 20, 20, 20, 20, 20
1160 data 20, 20, 20, 20, 20, 20, 20, 20, 20, 2
0, 20, 20, 20, 20, 20, 20
1170 data 20, 20, 20, 20, 20, 20, 20, 20, 20, 2
0, 20, 20, 20, 2A, 00, 0A
1180 data 20, 20, 20, 20, 20, 20, 20, 20, 20, 2
0, 20, 20, 52, 77, 61, 62
1190 data 73, 40, 6F, 6E, 20, 49, 6E, 73, 74, 6
1, 6C, 6C, 65, 64, 2E, 20
1200 data 20, 20, 20, 20, 20, 20, 20, 20, 20, 2
0, 2A, 00, 0A, 2A, 20, 43
1210 data 6F, 70, 79, 72, 69, 67, 68, 74, 20, 3
1, 39, 38, 37, 20, 44, 61
1220 data 6E, 20, 40, 6F, 6F, 72, 65, 20, 26, 2
0, 44, 61, 76, 69, 64, 20
1230 data 53, 60, 61, 6C, 6C, 6C, 2E, 20, 2A, 00, 0
A, 00, 00, 2A, 20, 52, 65
1240 data 73, 69, 64, 65, 6E, 74, 20, 69, 6E, 2
0, 52, 41, 40, 20, 6C, 6F
1250 data 63, 61, 74, 69, 6F, 6E, 73, 3A, 20, 2
0, 20, 20, 20, 20, 20
1260 data 20, 20, 20, 20, 20, 20, 2A, 00, 0A, 2
A, 20, 20, 53, 74, 61, 72
1270 data 74, 5A, 20, 24, 00, 00, 20, 65, 6E, 6
4, 5A, 20, 24, 00, 2E, 20
1280 data 20, 20, 20, 20, 20, 20, 2A, 00, 0A, 2
A, 20, 20, 20, 20, 20
1290 data 20, 20, 20, 20, 20, 20, 20, 20, 20, 2
0, 20, 20, 20, 20, 20
1300 data 20, 20, 20, 20, 20, 20, 20, 20, 20, 2
0, 20, 20, 20, 20, 20
1310 data 20, 20, 20, 2A, 00, 0A, 2A, 2A, 2A, 2
A, 2A, 2A, 2A, 2A, 2A, 2A
1320 data 2A, 2A, 2A, 2A, 2A, 2A, 2A, 2A, 2A, 2
A, 2A, 2A, 2A, 2A, 2A, 2A
1330 data 2A, 2A, 2A, 2A, 2A, 2A, 2A, 2A, 2A, 2
A, 2A, 2A, 2A, 2A, 2A, 2A
1340 data 2A, 00, 0A, 00, 41, F9, 00, 00, 02, 1
6, 23, C8, 00, 00, 00, 84
1350 data 4E, 75, 40, F9, 00, 00, 05, EA, 00, 7
C, 07, 00, 20, 79, 00, 00
1360 data 00, 9C, 22, 79, 00, 00, 04, A2, 30, 1
F, 33, 00, 23, 1F, 40, E1
1370 data 1F, 1F, 23, C9, 00, 00, 04, A2, 08, 0
0, 00, 00, 66, 02, 4E, 6F
1380 data 4E, 71, 30, 1F, 08, 50, 6C, 00, 03, 0

```

```

8, 0C, 40, 00, 04, 66, 00
1390 data 03, 66, 46, F9, 00, 00, 05, EA, 48, F
9, FF, FF, 00, 00, 03, 36
1400 data 33, 07, 00, 00, 00, 03, 2A, 23, EF, 00, 0
2, 00, 00, 03, 2C, 33, EF
1410 data 00, 06, 00, 00, 00, 03, 30, 33, EF, 00, 0
8, 00, 00, 03, 32, 33, EF
1420 data 00, 0A, 00, 00, 03, 34, 2E, 7C, 00, 0
0, 05, 76, 33, FC, 00, 20
1430 data 00, 00, 0A, E4, 33, FC, 00, 01, 00, 0
0, 0A, E5, 42, 00, 30, 39
1440 data 00, 00, 03, 34, 00, 7C, 00, 41, 61, 0
0, 06, B6, 61, 00, 04, 76
1450 data 30, 39, 00, 00, 03, 2A, 02, 40, 00, 0
1, 67, 22, 45, F9, 00, 00
1460 data 02, CE, 60, 20, 52, 44, 3A, 20, 0
0, 20, 57, 34, 3A, 20, 00
1470 data 20, 23, 36, 20, 00, 00, 20, 42, 55, 4
6, 3A, 20, 00, 00, 45, F9
1480 data 00, 00, 02, C8, 61, 00, 04, 2A, 36, 3
9, 00, 00, 03, 32, 61, 00
1490 data 04, 76, 45, F9, 00, 00, 02, D4, 61, 0
0, 04, 16, 36, 39, 00, 00
1500 data 00, 00, 00, 00, 04, 62, 45, F9, 00, 0
0, 02, D4, 61, 00, 04, 02
1510 data 26, 39, 00, 00, 03, 2C, 61, 00, 04, 3
6, 4C, F9, FF, FF, 00, 00
1520 data 03, 36, 60, 00, 02, 8E, 00, 00, 00, 0
0, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00
1530 data 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 0
1, 00, 00, 00, 02, 00, 00
1540 data 00, 03, 00, 00, 00, 04, 00, 00, 00, 0
5, 00, 00, 00, 06, 00, 00
1550 data 00, 07, 00, 00, 00, 00, 00, 00, 00, 00, 0
1, 00, 00, 00, 02, 00, 00
1560 data 00, 03, 00, 00, 00, 04, 00, 00, 00, 0
5, 00, 00, 00, 06, 00, 00
1570 data 00, 07, 00, 00, 00, 00, 00, 00, 00, 00, 0
1, 00, 00, 00, 02, 00, 00
1580 data 00, 00, 00, 00, 00, 04, 00, 00, 00, 0
5, 00, 00, 00, 06, 00, 00
1590 data 00, 07, 00, 00, 00, 00, 00, 00, 00, 0
9, 00, 00, 00, 0A, 00, 00
1600 data 00, 08, 00, 00, 00, 0C, 00, 00, 00, 0
0, 00, 00, 00, 0E, 00, 00
1610 data 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 0
1, 00, 00, 00, 02, 00, 00
1620 data 00, 03, 00, 00, 00, 04, 00, 00, 00, 0
5, 00, 00, 00, 06, 00, 00
1630 data 00, 07, 00, 00, 00, 00, 00, 00, 00, 00, 0
9, 00, 00, 00, 0A, 00, 00
1640 data 00, 08, 00, 00, 00, 0C, 00, 00, 00, 00, 0
0, 00, 00, 00, 0E, 00, 00
1650 data 00, 0F, 00, 00, 00, 00, 00, 00, 00, 00, 0
1, 00, 00, 00, 02, 00, 00
1660 data 00, 03, 00, 00, 00, 04, 00, 00, 00, 0
5, 00, 00, 00, 06, 00, 00
1670 data 00, 07, 00, 00, 00, 00, 00, 00, 00, 00, 0
9, 00, 00, 00, 0A, 00, 00
1680 data 00, 08, 00, 00, 00, 0C, 00, 00, 00, 00, 0
0, 00, 00, 00, 0E, 00, 00
1690 data 00, 0F, 00, 00, 00, 00, 00, 00, 00, 00, 0
1, 00, 00, 00, 02, 00, 00
1700 data 00, 03, 00, 00, 00, 04, 00, 00, 00, 00, 0
5, 00, 00, 00, 06, 00, 00
1710 data 00, 07, 00, 00, 00, 00, 00, 00, 00, 00, 0
9, 00, 00, 00, 0A, 00, 00
1720 data 00, 08, 00, 00, 00, 0C, 00, 00, 00, 00, 0
0, 00, 00, 00, 0E, 00, 00
1730 data 00, 0F, 00, 00, 00, 00, 00, 00, 00, 00, 0
1, 00, 00, 00, 02, 00, 00
1740 data 00, 03, 00, 00, 00, 04, 00, 00, 00, 00, 0
5, 00, 00, 00, 06, 00, 00
1750 data 00, 07, 00, 00, 00, 00, 00, 00, 00, 00, 0
9, 00, 00, 00, 0A, 00, 00
1760 data 00, 08, 00, 00, 00, 0C, 00, 00, 00, 00, 0
0, 00, 00, 00, 0E, 00, 00
1770 data 00, 0F, 00, 00, 00, 00, 00, 00, 00, 00, 0
1, 00, 00, 00, 02, 00, 00
1780 data 00, 03, 00, 00, 00, 04, 00, 00, 00, 00, 0
5, 00, 00, 00, 06, 00, 00
1790 data 00, 07, 00, 00, 00, 00, 00, 00, 00, 00, 0
9, 00, 00, 00, 0A, 00, 00
1800 data 00, 08, 00, 00, 00, 0C, 00, 00, 00, 00, 0
0, 00, 00, 00, 0E, 00, 00

```

[illegible]

2600 data 67,00,22,39,00,00,0A,DA,4E,7
 1,4E,71,02,00,00,00
 2690 data 00,FF,D2,00,20,41,24,3C,00,0
 0,00,0F,4A,79,00,00
 2700 data 0A,DE,67,00,24,3C,00,00,00,0
 7,4E,71,4E,71,4E,71
 2710 data 12,90,0A,79,00,00,0A,DE,67,0
 4,13,50,00,02,4E,71
 2720 data D3,FC,00,00,00,50,4A,79,00,0
 0,0A,DE,67,06,D3,FC
 2730 data 00,00,00,50,4E,71,D1,FC,00,0
 0,01,00,51,CA,FF,00
 2740 data 52,79,00,00,0A,E4,4E,71,4C,F
 9,FF,FF,00,00,0A,E0
 2750 data 4E,75,00,00,00,00,00,00,0
 0,00,00,00,00,00
 2760 data 00,00,00,00,00,00,00,00,0
 0,00,01,00,00,00,00,00,00,00,0
 2770 data 00,00,00,03,00,00,00,04,00,0
 0,00,05,00,00,00,06
 2780 data 00,00,00,07,00,00,00,00,00,0
 0,00,01,00,00,02
 2790 data 00,00,00,03,00,00,00,04,00,0
 0,00,05,00,00,06
 2800 data 00,00,00,07,00,00,00,06,00,0
 0,00,06,00,0A,12
 2810 data 1E,1C,06,01,9A,0E,0A,36,00,0
 6,00,00,00,00,06,00
 2820 data 00,00,12,0C,22,0A,0A,0A,0A,0
 A,0C,01,01,00,32,0E
 2830 data 00,00,08,26,AC,16,A2,82,1A,0
 6,06,06,06,06,06,06,06,06,0A,0
 6,06,06,06,06,06,06
 2840 data 06,06,06,06,06,06,06,06,0A,0
 6,06,06,06,06,06,06
 2850 data 06,06,06,06,06,06,0C,16,0A,0
 A,0E,08,0C,1A,20,0C
 2860 data 08,0C,18,0C,22,0C,06,00,1A,1
 6,14,1A,0A,00,00,00
 2870 data *

RWABS Monitor Checksums for Listing 1

100 data 01,544,391,421,536,534,560,00
 4,006,738,5495
 1050 data 015,777,896,746,837,751,600,
 725,027,028,7002
 1150 data 616,550,601,637,006,623,785,
 005,773,846,7130
 1250 data 733,601,690,622,564,550,771,
 032,033,757,7049
 1350 data 701,773,770,767,965,791,700,
 722,001,737,7011
 1450 data 665,725,701,652,728,719,047,
 553,474,490,6552
 1550 data 403,492,405,494,510,549,497,
 491,515,553,5077
 1650 data 501,495,519,557,505,492,516,
 554,502,496,5137
 1750 data 520,558,506,500,524,555,503,
 497,521,559,5243
 1850 data 507,501,525,563,511,490,522,
 560,773,806,5766
 1950 data 986,016,026,062,097,849,620,
 749,799,925,0529
 2050 data 757,806,837,069,053,035,777,
 046,3,902,7656
 2150 data 719,027,750,794,732,939,000,
 737,704,762,7060
 2250 data 704,840,761,014,023,764,026,
 721,806,730,7077
 2350 data 733,600,727,799,731,733,590,
 665,601,649,6916
 2450 data 724,217,522,315,300,699,069,
 964,775,736,6201
 2550 data 750,826,000,34,794,990,780,0
 17,966,011,7584
 2650 data 778,823,055,770,794,017,773,
 977,934,36,7557
 2750 data 540,485,501,494,503,536,690,
 721,730,590,5798
 2850 data 693,640,220,1561

Life Is Too Short To Waste Time Making Music.

Chances are, if you're using a sequencer, you're wasting a lot of time. Unless you're using RealTime™, the new sequencer from Intelligent Music.

RealTime lets you do everything as your music is playing. You can graphically edit to the finest detail. Instantly access all MIDI controls. Load and save Patterns, Songs and device setups. Even run other programs and desk accessories from within RealTime. All while you're listening.

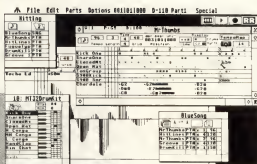
And to speed you up even more, RealTime includes interactive features. Like Track Bondage™, which lets you slave elements of different tracks to produce new combinations of musical elements. Automatic Fill Generation, which enables you to enhance your original material. Time Deviation, which allows you to give each track its own rhythmic feel. And much more.

So stop wasting your time making music. Get into RealTime today. Call or write us for the dealer nearest you. Or send a \$10 check and we'll send you a demonstration package. For the Atari 520, 1040, and MEGA ST.



Intelligent Music®
 116 North Lake Avenue
 Albany, NY 12206 USA
 Tel: (518) 434-4110
 Fax: (518) 434-0308

RealTime and Track Bondage are trademarks of Intelligent Computer Music Systems, Inc. Intelligent Music is a registered trademark of Intelligent Computer Music Systems, Inc. Atari and ST are trademarks of Atari Corp.



REALTIME™

RealTime features include:

- 256 simultaneous tracks
- 768 parts per whole-note clock resolution
- Device Lists to store drum machine and synth setups
- Copying, pasting, and editing of any region or Pattern
- Independent, nested track looping
- Graphic arrangement of Songs
- Import and export of MIDI Files
- Synchronization to SMPTE/MTC devices
- Complete GEM windows implementation

CIRCLE #105 ON READER SERVICE CARD.

**SUBSCRIBE
TO ST-LOG NOW!
FILL
OUT COUPON ON
PAGE 35...!**

END

C-MANSHIP

Listing 1: ST BASIC

```

000 OPEN"R:",w1,"A:MICROCHK.RSC",16:FILE
LDM1,16 AS BS
110 AS=AS+I*FOR I=1 TO 16:READ US:IF US=
"K" THEN LA0
20 ACVAL("XH")US:PRINT "X":AS=AS+CH
RUS
130 LET BS=AS+R+1:PUT 1,R:GOTO 10
140 CLOSE 1:PRINT:PRINT "ALL DONE!"
1000 data 00,00,0C,42,09,32,09,32,09,3
2,06,04,00,24,09,32
1010 data 06,64,19,C2,00,90,00,00,01,
0,00,00,00,00,00,00,00,00,00,00,00,00
1020 data 00,00,19,E2,20,20,20,20,4E,
4,5,57,20,41,43,43,4F
1030 data 55,4E,54,20,20,20,20,00,00,
0,20,20,20,20,20,20
1040 data 20,20,20,20,20,20,20,20,20,2
0,20,20,20,20,20,20,20,20,20,20,20,20
1050 data 20,20,20,00,4E,61,6D,65,3F,
20,5F,5F,5F,5F,5F
1060 data 5F,5F,5F,5F,5F,5F,5F,5F,5F,
5F,5F,5F,5F,5F,5F
1070 data 5F,5F,5F,5F,5F,00,58,58,58,58,
5,58,58,58,58,58,58,58,58,58,58,58,58
1080 data 58,58,58,58,58,58,58,58,58,5
8,58,58,58,58,00,20
1090 data 20,20,20,20,20,20,20,20,20,2
0,20,20,20,20,20,20
1100 data 20,20,20,20,20,20,20,20,20,2
0,20,20,20,20,20,20,20,20,20,20,20,20
1110 data 30,20,5F,5F,5F,5F,5F,5F,5F,
5,F,5F,5F,5F,5F,5F
1120 data 5F,5F,5F,5F,5F,5F,5F,5F,5F,
5,F,00,58,58,58,58
1130 data 58,58,58,58,58,58,58,58,58,58,
5,58,58,58,58,58,58,58,58,58,58,58,58
1140 data 58,58,58,58,58,58,00,20,20,2
0,20,20,20,20,20,20
1150 data 20,20,20,20,20,20,20,20,20,4
3,69,7,40,20,5F,5F,5F
1160 data 5F,5F,5F,5F,5F,5F,5F,5F,5F,
5,F,5F,5F,5F,5F,5F,5F,5F,5F,5F,5F,5F
1170 data 58,58,58,58,58,58,58,58,58,5
8,58,58,58,58,58,58,58,58,58,58,58,58
1180 data 20,53,74,61,74,65,3A,20,5F,5
F,00,41,41,00,20,20
1190 data 20,20,20,20,20,20,20,20,20,5
8,58,58,58,58,58,58,58,58,58,58,58,58
1200 data 5F,5F,5F,20,5F,5F,5F,5F,00,3
9,39,39,39,39,39,39
1210 data 39,39,00,20,20,20,20,20,20,2
0,00,20,42,61,6C,61
1220 data 6E,63,63,3A,20,24,5F,5F,5F,5
8,58,58,58,58,58,58,58,58,58,58,58,58
1230 data 39,39,39,39,39,39,39,4F,48,0
0,43,41,4E,43,45,4C
1240 data 00,20,44,65,73,68,20,00,20,2
0,65,6C,65,20,00,20
1250 data 43,68,68,63,68,73,60,00,20,5
0,00,62,69,68,68,68,68,68,68,68,68,68
1260 data 55,74,69,6C,69,74,65,69,73,0
0,20,20,41,62,6F,75
1270 data 74,20,40,69,63,72,6F,43,68,6
5,63,68,2E,2E,00
1280 data 20,20,20,20,20,20,20,20,20,2
0,20,20,20,20,20,20,20,20,20,20,20,20
1290 data 20,20,20,20,20,20,20,20,20,2
0,44,65,73,68,20,41
1300 data 63,63,63,73,73,6F,72,79,20,3
1,20,00,20,20,44
1310 data 63,63,68,20,41,63,63,65,73,7
3,6F,72,79,20,20,20,20,20,20,20,20,20
1320 data 20,00,20,20,44,65,73,68,20,4
1,63,63,65,73,73,6F
1330 data 72,79,20,33,20,00,20,20,20,4
6,65,73,68,20,41,63
1340 data 63,63,73,6F,72,79,20,34,2,0
0,20,00,20,20,20
1350 data 73,68,20,41,63,63,65,73,73,6
F,72,79,20,35,20,20
1360 data 00,20,20,44,65,73,68,20,41,6
3,65,65,73,73,6F,72
1370 data 73,68,20,20,20,00,20,4E,20,4
E,65,77,2F,2E,2E,00

```

1380 2a, 20, 4F, 20, 4F, 4F, 70, 65, 6E, 2E, 2E, 2E
 1390 2a, 20, 43, 20, 43, 6C
 1390 2a 6F, 73, 65, 2E, 2E, 2E, 00, 20, 20, 2D
 1400 2a, 20, 20, 20, 20, 20
 1400 2a 20, 20, 20, 20, 20, 4E, 20, 20, 00, 6F, 6E
 1410 2a 75, 2E, 2E, 2E, 2E, 2E, 2E, 2E, 2E, 2E
 1410 2a 40, 40, 20, 4E, 65, 77, 20, 40, 6F, 6E
 1420 2a 74, 6E, 2E, 2E, 00
 1420 2a 20, 45, 20, 45, 6E, 74, 65, 72, 2E, 2E
 1430 2a 53, 20, 53, 20, 53
 1430 2a 65, 61, 72, 63, 68, 2E, 2E, 2E, 00, 20
 1440 2a 53, 20, 53, 20, 53, 20, 53, 20, 53, 20
 1440 2a 6E, 63, 69, 6C, 65, 2E, 2E, 2E, 00, 20
 1450 2a 40, 43, 61, 6E, 63
 1450 2a 65, 6C, 2E, 2E, 2E, 2E, 00, 41, 20, 41
 1450 2a 74, 6F, 2E, 2E, 2E
 1460 2a 53, 20, 53, 20, 53, 20, 57, 69, 6E, 64, 6F, 7
 1460 2a 20, 00, 20, 47, 20
 1470 2a 52, 65, 67, 69, 73, 74, 65, 72, 00, 20
 1480 2a 20, 59, 20, 4E, 65, 77
 1480 2a 20, 59, 65, 61, 72, 2E, 2E, 2E, 2E, 00, 20
 1490 2a 44, 20, 44, 61, 74, 65
 1500 2a 40, 40, 40, 40, 40, 20, 49, 20, 49, 60, 7
 1500 2a 6F, 72, 74, 2E, 2E, 2E
 1500 2a 00, 4F, 48, 00, 43, 41, 4E, 43, 45, 4
 1500 2a 40, 20, 20, 20
 1510 2a 20, 00, 20, 41, 61, 60, 65, 3A, 20, 5F
 1510 2a 5F, 5F, 5F, 5F, 5F, 5F
 1520 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 58
 1530 2a 45, 20, 45, 49, 4E
 1530 2a 45, 20, 00, 00, 00, 4E, 65, 77, 20, 4
 1540 2a 41, 61, 74, 65, 00, 5F, 5F
 1540 2a 5F, 5F, 5F, 5F, 00, 5F, 5F, 2F, 5F, 5
 1550 2a 40, 40, 40, 40, 40, 40, 40, 40, 40, 40, 40
 1550 2a 39, 39, 39, 39, 39, 00, 4F, 48, 00, 43, 4
 1560 2a 43, 45, 4C, 00, 53
 1560 2a 45, 41, 52, 43, 48, 20, 58, 41, 52, 4
 1570 2a 45, 45, 45, 52, 53
 1570 2a 00, 40, 40, 4E, 4E, 54, 48, 53, 00, 4E, 5
 1580 2a 43, 43, 43, 43, 43, 43, 43, 43, 43, 43, 43
 1590 2a 41, 40, 4F, 55, 4E, 54
 1590 2a 53, 00, 20, 20, 00, 20, 46, 52, 4F, 4
 1600 2a 5F, 5F, 5F, 5F, 00, 5F
 1600 2a 00, 20, 00, 20, 00, 20, 54, 6F, 3A, 2
 1610 2a 5F, 00, 39, 39, 39
 1610 2a 20, 20, 20, 20, 00, 20, 46, 72, 6F, 6
 1620 2a 3A, 20, 5F, 5F, 5F, 5F
 1620 2a 00, 39, 39, 39, 39, 39, 00, 20, 20, 20, 2
 1630 2a 54, 6F, 3A, 20, 5F
 1640 2a 00, 20, 20, 20, 20, 20, 00, 39, 39, 39, 3
 1640 2a 20, 20, 20, 20, 20, 20, 20, 20, 20, 20, 2
 1650 2a 20, 20, 20, 20, 20, 20, 20, 20, 20, 20, 2
 1660 2a 20, 20, 20, 20, 20, 20, 20, 20, 20, 20, 2
 1670 2a 5F, 00, 39, 39, 39, 39, 39, 39, 39, 39, 3
 1680 2a 5F, 5F, 5F, 5F, 2E, 5F
 1690 2a 5F, 00, 39, 39, 39, 39, 39, 39, 39, 39, 3
 1700 2a 20, 20, 5F, 5F, 5F, 5F, 5F, 5F, 5F, 5F, 5
 1710 2a 5F, 5F, 5F, 5F, 5F, 5F, 5F, 5F, 5F, 5F, 5
 1720 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1730 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1740 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1750 2a 20, 20, 20, 20, 20, 20, 20, 20, 20, 20, 2
 1760 2a 65, 60, 6F, 3A, 20, 5F, 5F, 5F, 5F, 5F, 5
 1770 2a 5F, 5F, 5F, 5F, 5F, 5F, 5F, 5F, 5F, 5F, 5
 1780 2a 5F, 5F, 5F, 5F, 5F, 5F, 5F, 5F, 5F, 5F, 5
 1790 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1800 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1810 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1820 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1830 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1840 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1850 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1860 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1870 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1880 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1890 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1900 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1910 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1920 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1930 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1940 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1950 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1960 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1970 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1980 2a 58, 58, 58, 58, 58, 58, 58, 58, 58, 58, 5
 1990 2a 58, 58,

[illegible]

C-manship

```

6,00,00,11,80,00,00
2660 data FF,FF,00,03,00,0A,00,00,03,C
2,00,00,03,C5,00,00
2670 data 03,00,00,03,00,06,00,00,11,8
0,00,00,FF,FF,00,03
2680 data 00,08,00,00,03,00,00,00,03,D
5,00,00,03,E1,00,03
2690 data 00,06,00,00,11,80,00,00,FF,F
7,00,05,00,0C,00,00
2700 data 03,05,00,00,03,EB,00,00,03,F
5,00,03,00,06,00,00
2710 data 11,80,00,00,FF,FF,00,05,0,0
A,00,00,03,FA,00,00
2720 data 04,02,00,00,04,13,00,03,00,0
5,00,00,11,80,00,00
2730 data FF,FF,00,08,00,11,00,00,04,1
B,00,00,04,23,00,00
2740 data 04,32,00,03,00,06,00,00,11,8
0,00,00,FF,FF,00,08
2750 data 00,0F,00,00,04,3A,00,00,04,5
9,00,00,04,00,00
2760 data 00,06,00,00,11,80,00,00,FF,F
0,00,1F,00,27,00,00
2770 data 04,9F,00,00,04,BE,00,00,04,E
4,00,03,00,06,00,00
2780 data 11,80,00,00,FF,FF,00,1F,00,2
5,00,00,05,04,00,00
2790 data 05,AC,00,00,05,C6,00,03,00,0
6,00,00,11,80,00,00
2800 data FF,FF,00,08,00,1A,00,00,06,1
7,00,00,06,1C,00,00
2810 data 06,22,00,03,00,06,00,02,11,8
0,00,00,FF,00,05
2820 data 00,06,00,00,06,27,00,00,06,2
E,00,00,06,3D,00,03
2830 data 00,06,00,02,11,80,00,00,FF,F
F,00,07,00,0F,00,00
2840 data 00,00,00,06,63,00,00,06,8
9,00,03,00,06,00,02
2850 data 11,80,00,00,FF,FF,00,1F,00,2
6,00,00,06,A8,00,00
2860 data 06,80,00,00,06,8A,00,03,00,0
6,00,02,11,80,00,00
2870 data FF,00,08,00,0A,00,00,06,C
2,00,00,06,E1,00,00
2880 data 07,80,00,03,00,06,00,02,11,8
0,00,00,FF,FF,00,1F
2890 data 00,25,00,00,07,36,00,00,07,3
A,00,00,07,38,00,03
2900 data 00,06,00,02,11,80,00,00,FF,F
F,00,04,00,01,00,00
2910 data 07,3C,00,00,07,52,00,00,07,5
3,00,03,00,06,00,02
2920 data 11,80,00,00,FF,FF,00,16,00,0
1,00,00,00,CA,00,00
2930 data 08,04,00,00,08,05,00,03,00,0
6,00,02,11,80,00,00
2940 data FF,FF,00,0A,00,01,00,00,08,D
6,00,00,08,E5,00,00
2950 data 08,06,00,03,00,06,00,02,11,8
0,00,00,FF,FF,00,0F
2960 data 00,01,FF,FF,00,01,00,09,00,1
4,00,00,00,30,00,02
2970 data 11,22,00,00,00,00,00,25,00,1
1,00,02,FF,FF,FF,FF
2980 data 00,16,00,00,00,20,00,00,09,3
2,00,09,00,01,00,13
2990 data 00,01,00,03,FF,FF,FF,FF,00,1
E,00,08,00,00,00,00
3000 data 09,4E,00,02,00,03,00,21,00,0
1,00,04,FF,FF,FF,FF
3010 data 00,1E,00,08,00,00,00,09,6
A,00,01,00,05,00,23
3020 data 00,01,00,05,FF,FF,FF,FF,00,1
E,00,08,00,00,00,00
3030 data 09,06,00,00,07,00,17,00,0
1,00,06,FF,FF,FF,FF
3040 data 00,1E,00,08,00,00,00,09,A
2,00,04,00,09,00,00
3050 data 00,01,00,07,FF,FF,FF,FF,00,1
E,00,08,00,00,00,00
3060 data 09,0E,00,10,00,09,00,11,00,0
1,00,00,FF,FF,FF,FF
3070 data 00,1E,00,08,00,00,00,09,D
A,00,08,00,08,00,14

```

```

3080 data 00,01,00,09,FF,FF,FF,FF,00,1
A,00,43,00,20,00,00
3090 data 01,77,00,04,00,0E,00,00,00,0
1,00,00,FF,FF,FF,FF
3100 data 00,1A,00,61,00,20,00,00,01,7
A,00,14,00,0E,00,00
3110 data 00,01,FF,FF,00,01,00,08,00,1
9,00,00,00,00,00,00
3120 data 00,00,00,00,00,00,50,00,1
9,00,08,00,02,00,02
3130 data 00,1A,00,00,00,00,00,11,0
0,00,00,00,00,50
3140 data 02,01,00,01,00,03,00,07,00,1
9,00,00,00,00,00,00
3150 data 00,00,00,02,00,00,28,03,0
1,00,04,FF,FF,FF,FF
3160 data 00,20,00,00,00,00,00,01,8
1,00,00,00,00,00,06
3170 data 03,01,00,05,FF,FF,FF,FF,00,2
0,00,00,00,00,00,00
3180 data 01,88,00,06,00,00,00,06,03,0
1,00,06,FF,FF,FF,FF
3190 data 00,20,00,00,00,00,00,01,8
F,00,0C,00,00,00,08
3200 data 03,01,00,07,FF,FF,FF,FF,00,2
0,00,00,00,00,00,00
3210 data 01,98,00,14,00,00,00,09,03,0
1,00,02,FF,FF,FF,FF
3220 data 00,20,00,00,00,00,00,01,9
F,00,1D,00,00,00,08
3230 data 03,01,00,00,00,09,22,00,1
9,00,00,00,00,00,00
3240 data 00,00,00,00,03,01,00,50,00,1
3,00,12,00,0A,00,11
3250 data 00,14,00,00,00,00,FF,11,0
0,00,02,00,00,00,17
3260 data 00,08,00,00,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
3270 data 01,A0,00,00,00,00,17,00,0
1,00,0C,FF,FF,FF,FF
3280 data 00,1C,00,00,00,08,00,01,C
0,00,00,00,01,00,17
3290 data 00,01,00,00,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
3300 data 01,00,00,00,00,02,00,17,00,0
1,00,0E,FF,FF,FF,FF
3310 data 00,1C,00,00,00,00,00,01,E
0,00,00,00,03,00,17
3320 data 00,01,00,0F,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
3330 data 02,02,00,00,00,04,17,00,0
1,00,10,FF,FF,FF,FF
3340 data 00,1C,00,00,00,00,00,02,1
7,00,00,00,05,00,17
3350 data 00,01,00,11,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
3360 data 02,2C,00,00,00,06,00,17,00,0
1,00,09,FF,FF,FF,FF
3370 data 00,1C,00,00,00,00,00,02,4
1,00,00,00,07,00,17
3380 data 00,01,00,19,00,13,00,18,00,1
4,00,00,00,00,00,00
3390 data 11,00,00,08,00,00,10,00,0
6,00,14,FF,FF,FF,FF
3400 data 00,1C,00,00,00,00,00,02,5
6,00,00,00,00,00,10
3410 data 00,03,00,15,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
3420 data 02,60,00,00,00,01,00,10,00,0
1,00,16,FF,FF,FF,FF
3430 data 00,1C,00,00,00,00,00,02,6
8,00,00,00,02,00,10
3440 data 00,01,00,07,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
3450 data 02,77,00,00,00,04,00,10,00,0
1,00,18,FF,FF,FF,FF
3460 data 00,1C,00,00,00,00,00,02,8
8,00,00,00,05,00,10
3470 data 00,01,00,12,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
3480 data 02,90,00,00,00,03,00,10,00,0
1,00,1F,00,1A,00,1E
3490 data 00,14,00,00,00,00,FF,11,0
0,00,0E,00,00,00,00
3500 data 00,05,00,18,FF,FF,FF,FF,00,1

```

C-manship

```

C,00,00,00,00,00,00
3518 data 02,0A,00,00,00,00,00,10,00,0
1,00,1C,FF,FF,FF,FF
3520 data 00,1C,00,00,00,00,00,02,A
C,00,00,00,01,00,10
3530 data 00,01,00,1D,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
3540 data 02,89,00,00,00,03,00,10,00,0
1,00,1E,FF,FF,FF,FF
3550 data 00,1C,00,00,00,00,00,02,C
9,00,00,00,02,00,10
3560 data 00,01,00,19,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
3570 data 02,06,00,00,00,04,00,10,00,0
1,00,22,00,20,00,11
3580 data 00,14,00,00,00,00,00,FF,11,0
0,00,15,00,00,00,0C
3590 data 00,02,00,21,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
3600 data 02,E1,00,00,00,00,0C,00,0
1,00,1F,FF,FF,FF
3610 data 00,1C,00,00,00,00,00,02,E
D,00,00,00,01,00,0C
3620 data 00,01,00,00,00,23,00,25,00,1
4,00,00,00,00,00,FF
3630 data 11,00,00,1F,00,00,00,15,00,0
3,00,24,FF,FF,FF,FF
3640 data 00,1C,00,00,00,00,00,02,F
A,00,00,00,01,00,15
3650 data 00,01,00,25,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
3660 data 03,09,00,00,00,00,00,15,00,0
1,00,22,FF,FF,FF,FF
3670 data 00,1C,00,20,00,00,00,03,1
4,00,00,00,02,00,15
3680 data 00,01,FF,FF,FF,00,01,04,00,1
4,00,00,00,10,00,02
3690 data 11,22,00,00,00,00,00,17,00,0
9,00,02,FF,FF,FF,FF
3700 data 00,1A,00,43,00,20,00,00,03,2
1,00,02,00,05,00,08
3710 data 00,01,00,03,FF,FF,FF,FF,00,1
A,00,41,00,20,00,00
3720 data 03,24,00,00,00,05,00,08,00,0
1,00,04,FF,FF,FF,FF
3730 data 00,1E,00,08,00,00,00,00,09,F
6,00,04,00,03,00,0E
3740 data 00,01,00,00,FF,FF,FF,FF,00,1
6,00,20,00,20,00,00
3750 data 0A,12,00,00,06,00,01,00,0A,00,0
1,FF,FF,00,01,00,04
3760 data 00,14,00,00,00,30,00,02,11,1
2,00,00,00,00,00,17
3770 data 00,08,00,02,FF,FF,FF,FF,00,1
A,00,00,00,00,00,02
3780 data 03,55,00,02,00,01,00,13,00,0
1,00,03,FF,FF,FF,FF
3790 data 00,1D,00,08,00,00,00,00,0A,2
E,00,07,00,03,00,08
3800 data 00,01,00,04,FF,FF,FF,FF,00,1
A,00,43,00,20,00,00
3810 data 03,75,00,02,00,05,00,08,00,0
1,00,00,FF,FF,FF,FF
3820 data 00,1A,00,61,00,20,00,00,03,7
8,00,00,00,05,00,08
3830 data 00,01,FF,FF,FF,00,01,00,0E,00,1
4,00,00,00,30,00,02
3840 data 11,22,00,00,00,00,00,3A,00,1
1,00,02,FF,FF,FF,FF
3850 data 00,1A,00,00,00,20,00,00,03,7
F,00,06,00,01,00,2E
3860 data 00,01,00,03,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
3870 data 03,91,00,06,00,03,00,07,00,0
1,00,04,FF,FF,FF,FF
3880 data 00,1C,00,00,00,00,00,00,03,9
8,00,1D,00,03,00,07
3890 data 00,01,00,05,FF,FF,FF,FF,00,1
0,00,41,00,20,00,00
3900 data 03,AA,00,20,00,00,0E,0F,14,00,0
1,00,05,FF,FF,FF,FF
3910 data 00,0A,00,43,00,20,00,00,03,A
7,00,06,00,0E,00,14
3920 data 00,01,00,07,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
3930 data 03,AA,00,0C,00,06,00,07,00,0
1,00,08,FF,FF,FF,FF
3940 data 00,1E,00,08,00,00,00,0A,4
A,00,06,00,04,00,0A
3950 data 00,01,00,09,FF,FF,FF,FF,00,1
E,00,08,00,00,00,00
3960 data 0A,66,00,11,00,04,00,08,00,0
1,00,0A,FF,FF,FF,FF
3970 data 00,1E,00,08,00,00,00,0A,8
2,00,1D,00,04,00,0C
3980 data 00,01,00,0B,FF,FF,FF,FF,00,1
E,00,08,00,00,00,00
3990 data 0A,9E,00,2A,00,04,00,0A,00,0
1,00,0C,FF,FF,FF,FF
4000 data 00,1E,00,08,00,00,00,0A,8
A,00,0C,00,07,00,11
4010 data 00,01,00,0D,FF,FF,FF,FF,00,1
E,00,08,00,00,00,00
4020 data 0A,06,00,1E,00,07,00,0F,00,0
1,00,0E,FF,FF,FF,FF
4030 data 00,1E,00,08,00,00,00,0A,F
2,00,09,00,09,00,27
4040 data 00,01,00,00,FF,FF,FF,FF,00,1
E,00,28,00,00,00,00
4050 data 0B,0E,00,09,00,08,00,26,00,0
1,FF,FF,00,01,00,00
4060 data 00,14,00,00,00,10,00,02,11,2
2,00,00,00,00,00,35
4070 data 00,11,00,02,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
4080 data 05,03,00,14,00,01,00,00,00,0
1,00,03,FF,FF,FF,FF
4090 data 00,1C,00,00,00,00,00,05,1
1,00,0F,00,03,00,18
4100 data 00,01,00,04,FF,FF,FF,FF,00,1
A,00,43,00,20,00,00
4110 data 05,2A,00,05,00,0E,01,3,00,0
1,00,05,FF,FF,FF,FF
4120 data 00,1A,00,41,00,20,00,00,05,2
D,00,1D,00,0E,00,13
4130 data 00,01,00,06,FF,FF,FF,FF,00,1
4,00,08,00,00,00,FF
4140 data 11,00,00,01,00,04,00,34,00,0
7,00,07,FF,FF,FF,FF
4150 data 00,1A,00,11,00,00,00,00,05,3
4,00,03,00,05,00,08
4160 data 00,01,00,08,FF,FF,FF,FF,00,1
A,00,11,00,00,00,00
4170 data 05,3C,00,0F,00,05,00,08,00,0
1,00,09,FF,FF,FF,FF
4180 data 00,1A,00,11,00,00,00,00,05,4
5,00,1B,00,05,00,08
4190 data 00,01,00,0A,FF,FF,FF,FF,00,1
A,00,11,00,00,00,00
4200 data 05,4B,00,27,00,05,00,08,00,0
1,00,08,FF,FF,FF,FF
4210 data 00,1A,00,11,00,00,00,00,05,5
1,00,03,00,07,00,08
4220 data 00,01,00,0C,FF,FF,FF,FF,00,1
A,00,11,00,00,00,00
4230 data 05,55,00,0F,00,07,00,08,00,0
1,00,0D,FF,FF,FF,FF
4240 data 00,1A,00,11,00,00,00,00,05,5
A,00,1B,00,07,00,08
4250 data 00,01,00,0E,FF,FF,FF,FF,00,1
A,00,11,00,00,00,00
4260 data 05,5F,00,27,00,07,00,08,00,0
1,00,0F,FF,FF,FF,FF
4270 data 00,1A,00,11,00,00,00,00,05,6
6,00,03,00,09,00,08
4280 data 00,01,00,10,FF,FF,FF,FF,00,1
A,00,11,00,00,00,00
4290 data 05,7D,00,0F,00,09,00,08,00,0
1,00,11,FF,FF,FF,FF
4300 data 00,1A,00,11,00,00,00,00,05,7
8,00,1B,00,09,00,08
4310 data 00,01,00,12,FF,FF,FF,FF,00,1
A,00,11,00,00,00,00
4320 data 05,81,00,27,00,09,00,08,00,0
1,00,0D,FF,FF,FF,FF
4330 data 00,1A,00,31,00,00,00,00,05,B
A,00,15,00,09,00,08
4340 data 00,0A,00,FF,FF,00,01,00,04,00,1
4,00,00,00,10,00,02
4350 data 11,22,00,00,00,00,00,1F,00,0

```


C-manship

```

9,00,02,FF,FF,FF,FF
4360 data 00,1C,00,00,00,00,00,05,9
2,00,07,00,01,00,11
4370 data 00,01,00,03,FF,FF,FF,FF,00,1
D,00,00,00,00,00
4380 data 00,2A,00,03,00,03,00,19,00,1
1,00,04,FF,FF,FF,FF
4390 data 00,1A,00,43,00,20,00,00,05,C
E,00,03,00,06,00,08
4400 data 00,01,00,00,FF,FF,FF,FF,00,1
A,00,61,00,20,00,00
4410 data 05,01,00,11,00,06,00,08,00,1
1,FF,FF,00,01,00,00
4420 data 00,14,00,00,00,30,00,02,11,2
2,00,00,00,00,3A
4430 data 00,0E,00,02,FF,FF,FF,FF,00,1
C,00,00,00,00,00
4440 data 05,08,00,03,00,01,00,14,00,1
F,00,03,FF,FF,FF,FF
4450 data 00,1C,00,00,00,00,00,05,E
D,00,03,00,02,00,14
4460 data 00,01,00,04,FF,FF,FF,FF,00,1
C,00,00,00,00,00
4470 data 00,02,00,03,00,03,00,14,00,1
0,00,05,FF,FF,FF,FF
4480 data 00,1E,00,00,00,00,00,00,08,4
6,00,2E,00,01,00,00
4490 data 00,01,00,06,FF,FF,FF,FF,00,1
E,00,08,00,00,00
4500 data 00,52,00,25,00,03,00,11,00,1
0,00,07,FF,FF,FF,FF
4510 data 00,1E,00,00,00,00,00,00,08,7
E,00,03,00,06,00,27
4520 data 00,01,00,08,FF,FF,FF,FF,00,1
E,00,00,00,00,00
4530 data 00,9A,00,2B,00,06,00,08,00,1
0,00,09,FF,FF,FF,FF
4540 data 00,1E,00,00,00,00,00,00,08,B
6,00,04,00,08,00,26
4550 data 00,01,00,0A,FF,FF,FF,FF,00,1
A,00,43,00,20,00,00
4560 data 07,25,00,03,00,08,00,10,00,1
0,00,08,FF,FF,FF,FF
4570 data 00,1A,00,41,00,20,00,00,07,2
A,00,15,00,08,00,10
4580 data 00,01,00,0C,FF,FF,FF,FF,00,1
A,00,41,00,20,00,00
4590 data 07,2F,00,27,00,08,00,10,00,1
0,00,0F,FF,FF,FF,FF
4600 data 00,16,00,20,00,00,00,00,08,D
2,00,2E,00,08,00,03
4610 data 00,01,FF,FF,FF,00,01,00,13,00,1
4,00,00,00,30,00,02
4620 data 11,12,00,00,00,00,31,00,1
3,00,02,FF,FF,FF,FF
4630 data 00,16,00,00,00,20,00,00,08,E
E,00,0C,00,01,00,18
4640 data 00,01,00,03,FF,FF,FF,FF,00,1
C,00,00,00,00,00
4650 data 07,54,00,02,00,04,00,25,00,1
0,00,04,FF,FF,FF,FF
4660 data 00,1C,00,00,00,00,00,07,7
6,00,25,00,04,00,0A
4670 data 00,01,00,05,FF,FF,FF,FF,00,1
C,00,00,00,00,00
4680 data 07,81,00,02,00,05,00,21,00,1
0,00,06,FF,FF,FF,FF
4690 data 00,1C,00,00,00,00,00,07,A
3,00,21,00,05,00,0E
4700 data 00,01,00,07,FF,FF,FF,FF,00,1
C,00,00,00,00,00
4710 data 07,02,00,25,00,06,00,0A,00,1
1,00,08,FF,FF,FF,FF
4720 data 00,1C,00,00,00,00,00,07,8
D,00,02,00,07,00,21
4730 data 00,01,00,09,FF,FF,FF,FF,00,1
C,00,00,00,00,00
4740 data 07,0F,00,26,00,07,00,0A,00,1
0,00,0A,FF,FF,FF,FF
4750 data 00,1C,00,00,00,00,00,07,E
9,00,02,00,08,00,21
4760 data 00,01,00,08,FF,FF,FF,FF,00,1
C,00,00,00,00,00
4770 data 08,08,00,21,00,00,00,0E,00,1
0,00,0C,FF,FF,FF,FF

```

```

4780 data 00,1C,00,00,00,00,00,08,1
A,00,25,00,09,00,0A
4790 data 00,01,00,0D,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
4800 data 08,25,00,02,00,0A,00,21,00,1
0,00,0E,FF,FF,FF,FF
4810 data 00,1C,00,00,00,00,00,08,4
7,00,26,00,0A,00,0A
4820 data 00,01,00,0F,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
4830 data 08,51,00,02,00,08,00,21,00,1
0,00,10,FF,FF,FF,FF
4840 data 00,1C,00,00,00,00,00,08,7
3,00,26,00,08,00,00
4850 data 00,01,00,11,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
4860 data 08,7D,00,25,00,0C,00,0A,00,1
0,00,12,FF,FF,FF,FF
4870 data 00,1C,00,00,00,00,00,08,8
8,00,02,00,00,00,00
4880 data 00,01,00,13,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
4890 data 08,AA,00,25,00,00,00,0A,00,1
0,00,10,FF,FF,FF,FF
4900 data 00,10,00,63,00,20,00,00,08,8
5,00,14,00,10,00,00
4910 data 00,01,FF,FF,FF,00,01,00,02,00,1
4,00,00,00,30,00,02
4920 data 11,22,00,00,00,00,19,00,1
6,00,02,FF,FF,FF,FF
4930 data 00,1C,00,00,00,00,00,08,B
8,00,04,00,01,00,11
4940 data 00,01,00,00,FF,FF,FF,FF,00,1
6,00,20,00,00,00,00
4950 data 0C,0A,00,06,00,03,00,0C,00,1
1,FF,FF,00,01,0A,0A
4960 data 00,1A,00,30,00,02,11,2
2,00,00,00,00,00,28
4970 data 00,08,00,02,FF,FF,FF,FF,00,1
6,00,00,00,20,00,00
4980 data 0C,26,00,06,00,01,00,18,00,1
0,00,03,FF,FF,FF,FF
4990 data 00,1C,00,00,00,00,00,08,E
7,00,02,00,04,00,09
5000 data 00,01,00,04,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
5010 data 08,F1,00,0C,00,04,00,06,00,1
0,00,05,FF,FF,FF,FF
5020 data 00,1C,00,00,00,00,00,08,F
6,00,13,00,04,00,08
5030 data 00,01,00,06,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
5040 data 08,FF,00,18,00,04,00,0A,00,1
0,00,07,FF,FF,FF,FF
5050 data 00,1C,00,00,00,00,00,08,9
A,00,02,00,06,00,0A
5060 data 00,01,00,08,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
5070 data 09,15,00,00,06,00,06,00,1
0,00,09,FF,FF,FF,FF
5080 data 00,1C,00,00,00,00,00,09,1
A,00,13,00,06,00,08
5090 data 00,01,00,0A,FF,FF,FF,FF,00,1
C,00,00,00,00,00,00
5100 data 07,25,00,18,00,06,00,0A,00,1
0,00,00,FF,FF,FF,FF
5110 data 00,1A,00,27,00,20,00,00,09,2
E,00,0F,00,08,00,08
5120 data 00,01,00,00,0C,42,00,00,00,3
2,00,00,10,C2,00,00
5130 data 11,3A,00,00,11,82,00,00,13,1
A,00,00,14,E2,00,00
5140 data 15,5A,00,00,16,92,00,00,18,7
2,00,00,18,8A,00,00
5150 data *
```


C-MANSHIP Checksums for Listing 1

100 data 21,544,391,421,536,604,620,73
0,588,553,5008
1050 data 812,51,861,789,558,647,7,957
827,635,6144
1150 data 737,975,751,724,699,870,633,
890,758,697,7734
1250 data 756,811,842,883,780,724,794,
753,699,733,7775
1350 data 784,774,742,793,873,769,843,
752,791,818,7939
1450 data 821,726,785,760,837,692,854,
762,741,934,7912
1550 data 751,740,782,815,738,708,801,
666,730,860,7591
1650 data 706,845,677,575,670,35,30,84
4,805,574,5761
1750 data 595,20,73,869,851,768,870,84
8,812,821,6527
1850 data 815,823,897,840,778,916,876,
847,871,934,8597
1950 data 773,714,845,854,901,953,48,8
95,50,916,6949
2050 data 828,881,866,999,56,956,51,87
5,829,761,7102
2150 data 691,895,799,888,783,825,816,
742,806,908,8153
2250 data 862,784,857,759,797,842,853,
799,821,900,8274
2350 data 772,851,893,815,845,837,692,
739,742,746,7932
2450 data 812,742,522,605,556,747,644,
818,710,543,6899
2550 data 786,553,721,590,711,621,565,
616,592,668,6343
2650 data 656,802,776,690,716,715,789,
531,705,716,7096
2750 data 592,730,695,785,665,727,714,
577,723,582,6790
2850 data 790,661,822,740,591,699,578,
780,669,830,7160
2950 data 801,693,870,541,879,868,549,
849,882,586,7518
3050 data 854,924,615,868,893,592,659,

501,489,490,6885
3150 data 841,504,834,875,549,832,874,
562,499,518,6888
3250 data 579,869,921,570,867,921,586,
865,843,519,7540
3350 data 850,877,527,599,855,517,853,
857,537,866,7338
3450 data 875,539,856,582,601,872,904,
572,873,924,7598
3550 data 572,865,576,607,864,938,603,
601,880,595,7101
3650 data 866,862,530,677,860,544,874,
883,622,856,7582
3750 data 711,521,873,874,580,879,879,
593,701,881,7492
3850 data 592,865,892,582,887,931,621,
868,951,593,7782
3950 data 883,917,612,895,974,609,865,
971,610,854,8190
4050 data 725,503,846,859,535,860,881,
584,943,854,7590
4150 data 535,853,909,560,865,907,539,
863,913,576,7520
4250 data 868,926,555,853,906,574,851,
893,591,666,7683
4350 data 873,542,862,892,628,869,763,
529,871,915,7744
4450 data 598,856,857,583,871,890,589,
869,949,612,7674
4550 data 890,893,579,893,905,610,680,
858,628,859,7795
4650 data 882,571,864,890,682,862,939,
601,867,979,8057
4750 data 603,879,921,575,884,902,579,
882,893,588,7706
4850 data 867,943,581,872,969,618,685,
872,590,858,7855
4950 data 750,536,880,909,610,840,924,
583,845,963,7840
5050 data 539,850,879,540,862,879,578,
593,655,662,7837
5150 data 203,203

C-MANSHIP Listing 2: C

```
handle_messages ()
{
    switch ( msg_buf[0] ) {
        case MM_SELECTED:
            do_menu ();
            break;
        case WM_REDRAW:
            do_redraw ( (GRECT *) &msg_buf[4] );
            break;
        case WM_FILLED:
            do_full ();
            break;
        case WM_ARROWED:
            do_arrow ();
            break;
        case WM_USLID:
            do_uslide ();
            break;
        case WM_HSLID:
            do_hslide ();
            break;
        case WM_CLOSED:
            do_wind_close ();
            break;
    }
}

do_redraw ( rec1 )
{
    GRECT *rec1;
    (
        GRECT rec2;

        wind_update ( DEG_UPDATE );
        wind_get ( msg_buf[3], WF_FIRSTXVWH,
            &rec2.g_x, &rec2.g_y, &rec2.g_w, &rec2.g_h );

        while ( rec2.g_w && rec2.g_h ) {
            if ( rc_intersect ( rec1, rec2 ) )
                draw_interior ( rec2 );
            wind_get ( msg_buf[3], WF_NEXTXVWH,
                &rec2.g_x, &rec2.g_y, &rec2.g_w, &rec2.g_h );
        }
    )
}
```

```

    )
    wind_update ( END_UPDATE );
}

do_full ()
{
    if ( !full )
        wind_set ( w_h2, WF_CURRXYWH, fullx, fully, fullw, fullh );
    else
        wind_set ( w_h2, WF_CURRXYWH,
            fullx, fully, fullw, 316 - 162*(res==MED) );
    calc_vslid ( cur_count );
    full = !full;
}

do_menu ()
{
}

do_arrow ()
{
}

do_vslide ()
{
}

do_hslide ()
{
}

do_wind_close ()
{
}

draw_interior ( clip )
{
    GRECT clip;
    {
        GRECT r;
        int lines_avail, lines_shown;

        graf_mouse ( M_OFF, 0L );
        vswr_node ( handle, MD_REPLACE );
        if ( msg_buf[3] == w_h1 ) {
            if ( res == HIGH ) {
                r.g_x = 1;
                r.g_y = 337;
                r.g_w = 638;
                r.g_h = 63;
            }
            else {
                r.g_x = 1;
                r.g_y = 167;
                r.g_w = 638;
                r.g_h = 32;
            }
        }
        draw_rec ( r, 2, 4, GREEN );
        draw_buttons ();
    }
    else {
        if ( !full_draw )
            set_clip ( TRUE, clip );
        wind_get ( w_h2, WF_WORXXYWH, &r.g_x, &r.g_y, &r.g_w, &r.g_h );
        draw_rec ( r, 2, 8, WHITE );
        if ( full )
            lines_avail = 20;
        else
            lines_avail = 15;
        lines_shown = cur_count - cur_top;
        if ( lines_avail != lines_shown ) {
            cur_top = cur_count - lines_avail;
            if ( cur_top < 0 )
                cur_top = 0;
        }
        if ( cur_count == 0 )
            calc_vslid ( 1 );
        else
            calc_vslid ( cur_count );
        calc_hslid ( NUM_COLUMNS );
        update_chk_wind ();
        full_draw = FALSE;
        set_clip ( FALSE, clip );
    }
    graf_mouse ( M_ON, 0L );
}

draw_rec ( rec, inter, fill, color )
{
    GRECT rec;
    int inter, fill, color;
    {
        int pxy[4];

        graf_mouse ( M_OFF, 0L );
        vsf_interior ( handle, inter );
        vsf_style ( handle, fill );
        vsf_color ( handle, color );
        pxy[0] = rec.g_x;
        pxy[1] = rec.g_y;
        pxy[2] = rec.g_x + rec.g_w - 1;
        pxy[3] = rec.g_y + rec.g_h - 1;
        vr_rect ( handle, pxy );
        graf_mouse ( M_ON, 0L );
    }
}

```

```

set_clip ( flag, rec )
int flag;
GRECT rec;
{
    int pxy[4];

    pxy[0] = rec.g.x;
    pxy[1] = rec.g.y;
    pxy[2] = rec.g.x + rec.g.w - 1;
    pxy[3] = rec.g.y + rec.g.h - 1;
    vs_clip ( handle, flag, pxy );
}

update_chk_wnd ()
{
    int i, y;

    wind_get ( w_h2, WF_WORKXYWH, &wrkx, &wrky, &wrkw, &wrkh );
    i = cur_top;
    y = 0;
    while ( i < cur_count ) && ( i < cur_top + wrkh / charh ) {
        prnt_chk_wnd [ i, wrky + charh * y * charh ];
        ++i;
        ++y;
    }
}

prnt_chk_wnd ( index, row )
int index, row;
{
    char a[40], s[10];

    if ( left ) {
        v_gtext ( handle, 6, row, cur_chk_strc[index].cancel );
        v_gtext ( handle, 25, row, cur_chk_strc[index].number );
        sprintf ( a, "%5ld.X02ld",
            cur_chk_strc[index].amount/100, cur_chk_strc[index].amount*100 );
        v_gtext ( handle, 77, row, a );
        v_gtext ( handle, 169, row, cur_chk_strc[index].payee );
        strcpy ( a, cur_chk_strc[index].memo );
        a[24] = 0;
        v_gtext ( handle, 424, row, a );
    }
    else {
        v_gtext ( handle, 9, row, cur_chk_strc[index].payee );
        v_gtext ( handle, 264, row, cur_chk_strc[index].memo );
        strcpy ( s, cur_chk_strc[index].date );
        format_date ( a, s );
        v_gtext ( handle, 520, row, a );
    }
}

draw_buttons ()
{
    set_buttons ();
    button ( "BALANCE", bal_but, 35 );
    button ( "TRANS", trans_but, 131 );
    button ( "CHECKS", check_but, 227 );
    button ( "DEP", dep_but, 323 );
    button ( "MONTH", mnth_but, 419 );
    button ( "DATE", date_but, 515 );
}

set_buttons ()
{
    if ( balance < 0 && balance > (-100) )
        sprintf ( bal_but, "%5ld.X02ld", balance/100, labs(balance*100) );
    else
        sprintf ( bal_but, "%5ld.X02ld", balance/100, labs(balance*100) );
    sprintf ( trans_but, "%d", num_trans );
    sprintf ( check_but, "%d", num_chks );
    sprintf ( dep_but, "%d", num_deps );
    if ( month == -1 )
        strcpy ( mnth_but, "NONE" );
    else
        strcpy ( mnth_but, months[month] );
}

button ( str1, str2, x1 )
char *str1, *str2;
int x1;
{
    int x2, y1, y2;
    int pxy[10];

    x2 = x1 + 80;
    y1 = 174 * res;
    y2 = 190 * res;
    vsr_mode ( handle, MD_REPLACE );
    vsf_color ( handle, WHITE );
    pxy[0] = x1;
    pxy[1] = y1;
    pxy[2] = x2;
    pxy[3] = y2;
    v_bar ( handle, pxy );
    pxy[3] = y1;
    pxy[4] = x2;
    pxy[5] = y2;
    pxy[6] = x1;
    pxy[7] = y2;
    pxy[8] = x1;
    pxy[9] = y1;
    vsi_width ( handle, 3 );
    vsi_color ( handle, BLACK );
    v_pline ( handle, 5, pxy );
    center_butstring ( str1, x1, 184 );
    center_butstring ( str2, x1, 194 );
}

center_butstring ( str, x1, y )
char *str;
int x1, y;
{
    int x, x2;

    x2 = x1 + 80;
    x = ((x2-x1)-(strlen(str)*8))/2+1;
    v_gtext ( handle, x1+5, y * res, " " );
    v_gtext ( handle, x, y * res, str );
}

format_date ( d1, d2 )
char *d1, *d2;
{
    strcpy ( d1, d2 );
    d1[2] = '/';
    strcpy ( &d1[3], &d2[2] );
    d1[5] = '/';
    strcpy ( &d1[6], &d2[4] );
}

```

C-manship:

A Complete GEM Application, Part 2

END

ST-CHECK

A CHECKSUM PROGRAM FOR THE ST



BY CLAYTON WALNUM

ALL RESOLUTIONS

Typing in a BASIC program listing can be a frustrating and time-consuming task. Just one mistyped character will frequently render a program completely unusable. So to ensure that your program will run correctly, the entire listing must be checked character by character against the original. This can take many hours. To make matters worse, you can't trust your own eyes. Do you know how easy it is to overlook an O where a 0 is supposed to be?

Typing checkers like *ST-Check* take over the arduous task of proofreading your program files. Using this program can cut down your debugging time by a huge factor. When the checker's output matches that published with the listing, you can be sure your typing is accurate.

Introspection

When you run *ST-Check* against itself, you will get one of several results. The program may just give up and crash. In that case, go through the listing character by character until you find your typing error.

A second possibility is that the program will run okay, but will create all bad checksum data. This may indicate an error somewhere between Lines 80 and 420.

Find the typo and correct it.

The last possibility is that the checksum data will have only a few bad values. In this case, use the normal method detailed below to locate your errors.

Warning: Until you get your checksum data for *ST-Check* to match the data following the listing, you can't trust it to proofread other programs.

Using ST-Check

When you finish typing a *ST* BASIC program listing from the magazine, save a copy to your disk, and then run *ST-Check*. The program will first ask for a filename. Type in the name for the program you wish checked (the one you just saved to the disk), and press RETURN. You'll then be asked for a "bug" name. Enter a filename for the checksum file (this can be any name not already on the disk), followed by RETURN.

ST-Check will now proofread the program. When the checking process is complete, you'll have a file on your disk (saved under your bug name) which contains the checksum data for the program checked.

Check the last value of each line. If it matches the value in the published checksum data, go on to the next. If it doesn't match, you've got a typo.

To find the error, look at the line number of the data statement in which the bad value occurred. This number is equivalent to the first program line the data evaluates. Let's call this "Line X." Count the entries in the data line until you get to the bad value. We'll call this count "Y." Now look at the program you typed in. Starting with and including Line X, count down Y lines. The line you end up on will be the one containing the typo.

Correct the error, and then rerun *ST-Check*. When you get all the checksum data to match that published in the magazine, your new program is ready to run.

Passing the buck

Okay, friends. Here's where the truth comes to the fore. I can take only minimal credit for *ST-Check*, as it's virtually a direct translation from *D:CHECK2* (ANALOG #16) by Istvan Mohos and Tom Hudson. All accolades and tribute should be directed to those two fine gentlemen. I'm sure they'll divvy it up fairly, and perhaps pass a small share onto me. Thanks, guys!

You may now type in this month's *ST* BASIC program, secure in the knowledge that the searching eye of *ST-Check* is primed and ready.

ST-Check Listing 1 — ST Basic

```
10 ST CHECK typing validator by Clayt
on Hainum
20 based on a program by Istvan Mohos
and Tom Hudson
30 if peek($ystab)=1 then c1:=17 else c
1:=32
40 fullu 2:iclearv 2:gotox c1,0:?" "ST
CHECK":ex=0:sp=0:xc=0
50 input "Enter filename: ",fs:input "
Enter 0UG name: ",f15
60 on error goto 598:open "0",w1,f15:
open "1",w2,f15:close w2
70 open "1",w2,f15:open x goto 140,220
80 color 2:?" "Counting lines"lineco
unt=0:color 1
90 on error goto 578
100 line input2,15:linecount=linecount
+1
110 ? " ",f15:goto 180
120 close w2:q:=int((linecount/10)/dim c
(linecount)),p(q)
130 x:=1:goto 78
140 range=0:lyne=0:color 2:?"?:?" "Fill
ing array":color 1
150 ? " ",f15:count=0
160 line input2,15:is:count=count+1
170 lyne=val(is):r(range)=lyne:range=r
ange+1
180 on error goto 580
190 line input2,15:is:count=count+1:if c
ount=10 then 130
200 goto 198
210 close w2:xc:=2:goto 78
220 color 2:?"?:?" "Calculating checksu
ms":color 1
240 for i:=1 to linecount:checksum:=0:li
ne input w2,15:line(i)
245 if mid$(is,1,1)="" then i:=1:igot
o 245
250 for z:=1 to 1:number=asc(mid$(is,z,
1))
260 if number=asc(" ") and ex=0 and sp
=1 then goto 328
270 if number<asc(" ") then sp=0 else
sp=1
280 if number<>34 then 308
290 if ex=1 then ex=0 else ex=1
```

```
300 if ex=0 and number=asc("a") and n
umber<asc("z") then number=number+32
310 product:=number:checksum=checksum
+product:xc:=1:if xc=4 then xc:=1
320 next z:?" "
330 checksum=checksum-1000*int(checksum
/1000):c(i)=checksum:z:=next i
340 close w2:lyne=r(0):item=0
350 color 2:?"?:?" "Creating BUG file":
color 1
360 count:=0:total:=0:if linecount<10 t
hen count=linecount
370 is:=str$(lyne):is:=is+" data "
380 for i:=1 to count:datum:=c(i*item+1)
390 is:=is+str$(datum):is:=is+" ",":total
:=total+datum:next i
400 is:=is+str$(total):print w1,is:?" "
410 item:=item+1:linecount=linecount-1
420 if linecount<1 then 438
430 lyne=r(item):goto 360
440 ? "To check 0UG data against the c
hecksum data found in the magazine,"
450 ? "return to the GEN desktop and d
ouble click your 0UG file. You may"
460 ? "then SHOW the data on your scre
en or PRINT the data to your printer,"
470 ? "The line number of each data st
atement coincides with the first line"
480 ? "of the user program the data st
atement evaluates. Numbers within"
490 ? "each data statement represent c
onsecutive lines of the user program."
500 ? "The last number is the total.":
? "
510 ? "Check the last number of each s
tatement against the version in the"
520 ? "magazine. Only when there's a
discrepancy need you check each number"
530 ? "in the data statement.":?
540 ? "Take note of the lines containi
ng typos, then make corrections. When"
550 ? "all corrections have been made,
re-run this program to double check."
560 ? "Press (RETURN)":input i:close
w1:close w2:end
578 if err=62 then resume 128
588 if err=62 then resume 218
```

```
598 if err=53 then ? chr$(7):"FILE NOT
FOUND!":close:resume 58
600 ? "ERROR #":err):" at LINE "ier:ien
d
```

ST-Check Checksums

```
18 data 447, 129, 283, 510, 661, 160
, 942, 482, 640, 556, 4730
, 118 data 25, 905, 797, 52, 79, 349,
832, 544, 9, 482, 4114
218 data 883, 479, 834, 822, 42, 498
, 255, 165, 826, 410, 5214
318 data 337, 1, 166, 578, 136, 801,
898, 937, 271, 769, 4894
418 data 363, 99, 153, 889, 243, 764
, 168, 192, 986, 156, 3935
518 data 757, 251, 146, 589, 146, 91
6, 539, 541, 733, 845, 5383
```

ST CHECK

END

BOOT UP TO BIG SAVINGS!



12 Issues \$28

\$19 OFF THE COVER PRICE

12 Issues with Disk \$79

NEW LOWER PRICE

The world of ATARI-ST continues to grow by leaps and bounds, and ST-LOG is there every step of the way!

- ☐ 12 Issues \$28
MCUWW
- ☐ 12 Issues with Disk \$79
DCUWW



☐ PAYMENT ENCLOSED ☐ BILL ME
☐ CHARGE MY: ☐ VISA ☐ MASTERCARD

CARD #	EXP	SIGNATURE
NAME		
ADDRESS		
CITY		STATE ZIP

MAKE CHECK PAYABLE TO L.P.P., INC., P.O. Box 10936, N. Hollywood, CA 91618. Offer Expires Nov. 29, 1989

Turbo ST

SofTrek
P.O. Box 5257
Winter Park, FL 32793
(407) 657-4611

Reviewed
by
David Plotkin



- Speeds computer response; saves time, reduces fatigue
- Easy-to-use desk accessory, automatically installs
- Works on any 520, 1040, or Mega ST with TOS in ROM

I've always thought that my ST redraw its screen fairly quickly. Oh, sure, you could see it being done (none of that "blink of the eye" stuff), but it was adequate. SofTrek, however, thought differently, and the result was *Turbo ST*. This program increases the speed of screen redraws by a sizable factor. For example, GFA BASIC is sped up by over 50%. When you consider that the blitter chip only offers a speed increase of about 20%, *Turbo ST*'s performance is impressive.

Now, given the fact that *Turbo ST* is strictly software, and the blitter is a hardware chip supposedly designed to speed all sorts of block moves in memory, how could *Turbo ST* be that much better? The answer comes in two parts. First of all, the blitter is intended to move all sorts of memory around faster, not just screen memory. *Turbo ST*, however, is designed to do just one thing—manipulate screen memory. It is optimized for that task, and for that single purpose it works well. The second part of the answer is that it's fairly well known that the ST's operating system is slow. It's written primarily in C, and the code is not particularly efficient. Many are the power-programmers, who claim that if the OS were rewritten in assembly (machine) language, it

would be much faster. And that is exactly what Wayne Buckholdt at SofTrek did: *Turbo ST* is written completely in 68000 machine language.

Installing *Turbo ST* is simple; it's a desk accessory you must copy to your boot-disk root directory. It will even install within CodeHead's *MultiDesk*. Once *Turbo ST* is active, you can turn it on or off just by clicking on the entry under the desk menu to activate a dialog box.

The effect of *Turbo ST* is apparent immediately when you open windows on your desktop. It also noticeably increases the speed of programs, such as GFA BASIC, dBMan, Data Manager and ST Writer. There are two things to remember, however, before you run right out and buy this product. First of all, it only increases the screen-drawing speed of programs that use text. Programs that draw each letter on the screen (such as *Easy Draw* or *WordUp*) or that deal exclusively in graphics will not benefit from *Turbo ST* at all. The other caveat is that *Turbo ST* doesn't always get along with other programs, although this is becoming a rarer occurrence as the product is improved. For example, Version 1.3 did not allow *TimeWorks' Publisher ST* to print. Ver-

sion 1.4 fixes this problem if you use CodeHead's *G+Plus* (with a supplied patch) instead of GDOS. However, Version 1.4 doesn't get along with MiGraph's new *TouchUp*, causing the rotate function for graphics to misbehave.

Should you buy *Turbo ST*? That depends. If you do a lot of text work and would appreciate the screen speed-up to the tune of \$50, then by all means, go ahead. It does work, and apart from an occasional incompatibility, it works well. However, if you aren't bothered by the ST's standard screen redraw or don't use text applications much, then save that \$50 for something else. ■



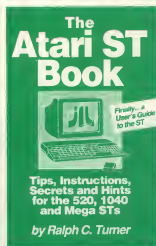
David Plotkin has been pounding the keys on Atari computers for almost ten years now. In that time he has written many memorable programs and articles that have been published in a variety of magazines. He holds an MS in Chemical Engineering and is a data analyst for Chevron Corporation.

The Atari ST Book

by Ralph C. Turner

Index Legalis Publishing Co.
P.O. Box 1822-38
Fairfield, IA 52556
(515) 472-2293
150 Pages; \$16.95

Reviewed
by
David
Plotkin



The Atari ST Book is a short, well-written volume full of hints, tips and detailed instructions for getting the most out of your ST. The book is written on a novice level, so it doesn't cover many advanced topics, such as programming. However, it does a nice job with the topics it does cover.

TOPICS COVERED INCLUDE:

- RAM Disks •
- Using (or Replacing) the Item Selector
- The DESKTOP.INF File and the Control Panel
- Desk Accessories •
- Installing an Application •
- Printers and How to Control Them •
- Word Processors •
- Desktop Tricks •
- Files and Their Management •
- The UTO Folder •
- Copying •
- Adjusting the Control Panel's Palette Settings
- Cold Boots and Warm Boots •
- Public Domain Software •
- Modems and Telecommunications and many other subjects •

The book is arranged into a variety of more-or-less unrelated topics, with each one the subject of a chapter. The organization is a little odd and disjointed. For example, the first chapter is about RAM disks and their advantages. From there, the book goes on to discuss DESKTOP.INF files and the AUTO folder in the context of getting your RAM disk installed. Chapter Two is about files and their management, including the types of files, folders, directories, filenames, wild cards and how to navigate around your disks.

A detailed description of the item selector is discussed in Chapter Three, and although the book is dated (as any book will be), Mr. Turner offers specific advice relat-

ed to products such as the *Universal Item Selector* (Version 1). Even though Version 2 of this fine product has since been released, the information in the book is still valuable.

Chapter Four takes up once again the subject of the AUTO folder, and the succeeding chapters discuss copying (including copy protection and backup products), the control panel, desk accessories and modifying the DESKTOP.INF file.

The book then goes on to discuss public-domain software and installing applications. Again, much of the information provided about available programs is unavoidably dated (there are now better programs than the ones cited), but still interesting.

Printers are the subject of the next chapter, which deals with the differences between parallel and serial, as well as dot-matrix versus daisy-wheel printers. Admittedly, this is basic stuff, but if you don't know it, the material is presented in a personal, friendly fashion. This chapter soon moves on to more complex material; for example, how to set the printer's special features.

Also included is a chapter on what must be the most confusing aspect of computers: modems and telecommunications. The entire chapter is presented in a question-and-answer format, which makes it a little hard to follow; but again, there is a lot of good information here.

The chapter on word processors discusses many of the features that different word processors have, as well as expounding Mr. Turner's opinion of what constitutes a desirable feature. Of course, since word processors all do things differently, this section is not very useful (you'll still need to read the manual that came with your word processor), but at least

it gives a fair indication of what to look for before buying.

Specific available word processors are covered, with discussion of the strong and weak points about each. Oddly, although Mr. Turner included such offerings as *Microsoft Write* and *Regent Word II* (which have hardly taken the ST world by storm), he left out such strong contenders as *WordUp* and *Timeworks' Word Writer ST* (likely the most popular of commercial word processors for the ST).

The next-to-last chapter handles miscellaneous subjects, covering a wide range, from selecting an item in an inactive window to what to do about hooking up two monitors. Mr. Turner's opinions tend to be somewhat biased. For example, he suggests buying the Practical Solutions *Monitor Master*. Fair enough, but what about competing products, such as *Astra's SW2*? One of the shortcomings of this book is that when discussing products for purchase, it mentions only a few (and not always the best) of the choices available.

The last chapter, on Chess, is less than two pages long and gives a brief overview of two chess programs available for the ST. The appendices list the names of many magazines for the ST, although a large number of these are no longer in publication.

Overall, this is a good book, especially for beginners. An experienced hand (like yours truly), however, will be surprised at how much new information is here; many of the tips and shortcuts are handy. The organization could be better, and the switching between narrative style and question-and-answer style chops up the book. The illustrations are amateurish, featuring handwritten notes on screen dumps. To be fair, though, they do get the message across.

You should check this book out. ■

3½-inch Dual Drives

IB Computers, Inc.
1519 S. Marlow
Portland, OR 97225
(503) 297-8425
\$320

**Reviewed
 by
 Pamela Rice Hahn**



Almost two years ago, I replaced one of my single-sided (SS) ST drives with a double-sided, double-density (DS/DD) drive. As time went on, I found myself getting more and more frustrated each time I'd get the "Drive B doesn't respond..." window, finding that I had once again erroneously inserted a DS/DD-formatted disk into my single-sided disk drive. I rationalized that while it would be time-consuming to transfer those zillions of files from SS disks to DS, the purchase of an additional DS disk drive wasn't really going to cost me all that much if I considered the extra disk capacity I'd have available. Once I'd convinced myself that this was the right move, I set out to convince my husband, Carl.

Carl agreed that something needed to be done. After some discussion on the subject, we

ruled out adding a hard drive because, although I envy the 20-megabyte Supra Carl bought a year ago, I would still have the incompatible disk formats. So, I added a hard drive to my wish list and started to research the DS drives available.

Looking through some back issues of user-group newsletters, I found something of interest written by MVACE (Miami Valley ACE, Huber Heights, OH) Secretary-Treasurer Marty Freed. He described IB Computer's dual-drive setup.

Carl has a Shanner drive, which is a similar setup, housing the two drives in one case. Shortly after Carl purchased his, we had looked into getting a Shanner dual drive for me as well, but Shanner had since gone out of business. Because the IB drives only cost about

\$100.00 more than just adding an Atari DS/DD or GTS-100, we decided to go ahead with the IB drives and keep our Atari SF354 as a back-up drive, should any of our units ever have to go in for service.

After placing a call to IB Computers, here's what I eventually (more on this later) got for \$322.38:

- 2—Chionon ½ height, 3.5-inch 720K disk drives
 - 1—metal case with power supply/cable
 - 1—data cable with two 34-pin edge connectors
 - 1—strain-relief plate
 - 12—flathead Phillips screws
 - 12—short flathead Phillips screws
 - 2—roundhead screws
- Absolutely no instructions. Assembly, however, was not all that difficult. I simply handed the

entire bundle over to my husband and asked him to do it. All told, it probably took him about an hour to complete the job.

The case and the power cord are already assembled. Next, because the drives both come preset as "Drive A," you must reconfigure one as Drive B. With the drive turned over so that the electronics board is facing you and the connector is up, there are three rows of five pins each, with two small black shorting plugs just below the connector. The Drive A configuration has one in the upper left connecting rows 1 and 2, with the other plug in the lower right connecting rows 2 and 3. When the lower right plug is moved over one column to the right while still being connected to rows 2 and 3, the drive is configured as Drive B.

Both 34-pin edge connectors on the data cable are wired identically. Connect the bottom (Drive B) drive's data connector and 34-pin edge power connector. Then insert the four flathead screws for Drive B into the bottom of the case and align and secure Drive B to the case.

Next, attach the data connector and power edge connector to the upper drive and set the drive on top of the lower drive. Route the data cable around or over the power supply, and then place it onto the cutout on the back of the metal case. Place the strain-relief plate over the top of the cable and attach it with the two roundhead screws. The upper portion of the case is now placed over the drives and attached by inserting four flat-head screws on each side of the case. (It is important to align the screw holes, but not difficult since the prior positioning of the lower drive has already set the alignment.)

Finally, insert the two small flat-head screws into the top of the metal case in back of the vents. Now, as soon as you connect the drives to your computer, you are in business.

Carl called IB Computers to confirm the above-described alignment, and while we did not get the name of the technical support representative he talked to, he was

courteous and helpful in answering Carl's questions.

After almost a year, my drives continue to work flawlessly. I'm not one for getting into computer and peripheral performance/benchmark tests, but so far my drives have done all I have asked of them. They work with every copy program I've tested them with (i.e. ProCopy, X-Utility, etc.). They've accepted my requests to format disks up to 800+K.

The IB service representative also assured us that while some Atari drives made by Chinon will read but not write to tracks 81 and 82, those drives are made to Atari specifications, not Chinon's standard drive specs. Because the IB drive arrangement uses the unaltered Chinon drives, they will read and write to those tracks, should you choose to do so. I am now convinced that I lose some disk reliability when formatting my data disks to that capacity and now choose to limit my disks to 80 tracks with ten sectors per track; however, some copy-protection schemes involve those final two tracks. If you want to back up those disks, you must have a drive that will write to those tracks.

Also, the drives are relatively quiet. The first week or so I had them in operation, whenever I'd first power up the drives the initial head rotation(s) created a sound somewhat similar to a migrating goose somewhere off in the distance. Now there's simply a quick, low whooshing noise and then silence.

I'm pleased with my purchase. One of my only two complaints pertains to IB Computers themselves. (I had originally ordered my drives on a Thursday, hoping to have them by the following weekend. The gentleman who took my order explained that since the drives wouldn't be shipped until the first day of the week, I should spring for the extra four bucks for UPS Blue Label shipping, which would ensure I'd receive them the second day after shipment. Considering the distance between Oregon and Ohio, that seemed reasonable. Two weeks later I was still waiting for my disk drives. I called IB, only to be informed

they had inadvertently set my order aside, apologized for their oversight and then processed my order; I finally received the drives two days after that.

My other complaint deals with the fact that I was not informed there would be a VISA surcharge. I always try to obtain the final invoice amount total so I can keep immediate records to verify against my VISA bill. The gentleman taking my order was vague, stating something about the uncertainty of what the final package would weigh and that he needed that weight to compute the shipping charges, etc. He convinced me, because I hung up the phone only knowing approximately what my drives were going to cost. Now, in hindsight, I can't help wondering whether they were trying to bury the VISA surcharge.

While I wholeheartedly endorse the drives, I do fault IB's customer relations. Anyone with a credit card knows they occasionally pay a surcharge for that convenience, but in this case, having that surcharge tacked on along with paying an additional Blue Label

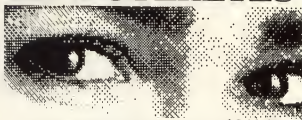
shipping charge for a set of drives it took me 2½ weeks to receive has me a little upset.

If I had it all to do over again, would I order the same drives from the same company? You bet. We received technical phone support when we needed it, and the drives are great. I probably just placed my order with a rookie employee. Maybe I'll eventually find it in my heart to forgive IB. We'll see. . . . ■



Pamela Rice Hahn has been an Atari enthusiast for five years. An active member of MVACE and the former editor of The TCAUC Newsletter, she continues to dispatch at a small-town police department to help support her family's computer habit, all the while dreaming of someday fulfilling her desire to become a full-time freelance writer.

COMPUTEREYES™



Capture Color Video Images: only \$249.⁹⁵

Capture B/W Images: only \$149.⁹⁵

Now you can easily and inexpensively acquire images from any standard video source — videotape, camera, disk — for display on your ST. Capture time is less than six seconds, and you have complete control of displayed color levels (or gray scales in monochrome).

ComputerEyes has everything you need: Interface hardware, complete easy-to-use software support on disk, and owner's manual. It fully supports all ST graphics modes and a variety of image formats and is compatible with virtually all popular graphics programs.

Think of the possibilities! ComputerEyes is backed by a one year warranty and the success of over 10,000 systems sold. Satisfaction guaranteed or return it within ten days for full refund. Demo disk available, \$3. See your dealer or order direct. For more information call 617-329-5400.

To order call 800-346-0090
or mail your order to: Digital Vision, Inc.
66 Eastern Avenue, Needham, MA 02456
VISA, M/C, or C.O.D. accepted
\$4.95 for ComputerEyes
Mass. residents
add 5% sales tax

**DIGITAL
VISION**

CIRCLE #106 ON READER SERVICE CARD.

Manhunter: New York

Sierra On-Line, Inc.
P.O. Box 485
Coarsegold, CA 93614
(209) 683-6858
\$49.95, Color or Monochrome

Reviewed
by
Peter A. Smith

You are roused from slumber by the specter of a huge floating eye. "Attention Manhunter!" it says. "There has been an explosion at Bellevue Hospital! Investigate." With that, it leaves as silently as it arrived. You yawn, blink the sleep from your eyes and retrieve your Manhunter Assignment Device (MAD). "So," you mumble to yourself, "this is the life of a Manhunter."

Thus begins *Manhunter: New York*, from Sierra On-Line. It seems the Earth has been invaded by an alien race called the Orb Alliance. Their rule is iron-fisted, and you have been chosen to be their servant. All humans have been implanted with a tracking device, but the device's design is faulty; it loses its signal if the tracked subject goes underground. Therefore, the Orbs employ Manhunters such as yourself to investigate suspicious actions by your fellow humans.

Your job is to identify the perpetrators of various crimes. You will be hindered by your fellow humans, as well as the Orbs in some cases. Death in this world comes swiftly

and frequently—most often at the hands of your fellow man. Your MAD will provide valuable assistance in tracking and identifying suspects.

Generally, the first thing to do after being given an assignment is to consult the tracking module of MAD. This will show you where your suspect has been recently. Your job is to trace his path and determine what he has been up to. If you can learn his name, the MAD info module will give you pertinent data, including his address, and access to his apartment. Should the trace show contact with another human, you may switch the tracker to this new suspect. Early in the game your choices of where to go are limited, but as you learn more, the Orbs will allow you to travel to a greater variety of locations in the course of your work.

Sierra is most famous for its *King's Quest* series, featuring a novel adventure-game design in which you use a joystick to move your on-screen persona, which you see in a third-person perspective. *Man-*

hunter is in some ways more traditional in that most of the game is viewed through the eyes of your alter ego. There are some third-person scenes, primarily of yourself getting slain in some exotic ways, but these are usually decorative in nature.

There is little text input involved, only the occasional typing of a name or brief phrase. Generally, whatever is visible to the Manhunter will be shown on your screen, along with a cursor. Moving this cursor over different parts of the screen will cause it to change shape. These shapes allow you to choose from available options, i.e., a magnifying-glass shape allows a closer look at something, an arrow shape allows you to move in the direction of the arrow.

Sierra has gone out of its way to make the game easy to use. You have the choice of mouse, joystick, cursor keys or numeric keyboard to move the on-screen cursor. The various special commands (save, restore and the like) can be called either with function

keys or drop-down menus.

The drop-down menu bar is not usually visible; a press of the right mouse button will bring it into view. Unfortunately, there is a bug in this segment of the program. Several times the bar would start popping into view, flashing drop-down menus on and off. While a definite distraction, this bug is not serious, and a tap or two on the escape key fixed the display.

Another convenient feature Sierra has included is demonstrated each time you are killed. A picture of *Manhunter: New York's* creators appears with a dialogue balloon offering advice. Then they "...take you back to a few moments before you made your fatal error."

Manhunter will run on either color or monochrome systems, although I'd recommend the color. Not only is it more visually pleasing, but it's easier to see small items lying about that may have significance. The game comes on five single-sided disks (Sierra has stated that future ST programs will come only on double-sided disks, much to the dismay of many 520ST owners) and is keyword copy protected (you must type in a word from the manual in order to load the game). Kudos to Sierra for using this style of copy protection.

In addition, a program is provided that will automatically install *Manhunter* on your hard disk. Again, thank you Sierra! It's about time companies started realizing the prevalence of hard-disk users among the gaming audience. If you don't have a hard disk and are apprehensive about shuffling five disks in and out of your disk drive, be assured that disk swaps are infrequent and unobtrusive.

Manhunter: New York is intended for mature audiences. Parents should consider whether their younger, impressionable children should play this game. I was raised in the old days, before they took all the violence out of Saturday-morning cartoons. I've yet to hatchet-murder anyone, but I still feel it is my duty to warn you; this game is violent, and in some places downright gory. (Early in the game there is a scene where a brawny bartender grabs you by the head, places his thumbs on your cheeks and squeezes un-



MANHUNTER: NEW YORK



til your head explodes. The scene then zooms out to show a glob of gore flying up toward the ceiling.)

Being an adventure wimp, I sometimes make it a habit to log on to telecommunications services and download hints on whatever game I'm playing, sometimes before I've really started playing it. (Taboo, I know, but I hate becoming utterly frustrated before I become involved in the game.) I did this with *Manhunter*, and it was a mistake. The game was not as difficult for me as I thought. Most of the puzzles had commonsense solutions, more so than I've found to be true in the majority of adventure games. To me this was a plus, but I think it's really a matter of personal preference. One facet of adventure gaming that I certainly did not miss is the guess-the-right-word syndrome that plagues many text and text/graphic adventures.

Several arcade sequences are embedded in the game. One arcade sequence in particular annoyed me. It took me over three hours to get past it. To say I was frustrated would be an understatement. If I want a difficult arcade game, I'll go out and buy one, thank you very much. If I buy an adventure game, I want puzzles to solve and a good plot to enjoy, not an eye-to-hand coordination workout. Still, I must admit, most of the other arcade sequences were interesting diversions, and more importantly, provided hints that were vital to completing the game.

One other glitch (besides the menu-bar bug) occurs when you

visit Coney Island. (Hey, even a *Manhunter* needs a little R & R.) If you attempt to use your MAD at this location, the game will crash with two or three bombs. Be assured you don't need your MAD at this point, so just don't use it. (I have duplicated this bug on both a Mega ST4 and a one-meg 520ST, the latter being a totally "clean" system.)

I enjoyed *Manhunter: New York*, and am happy to see that a sequel is planned. The grapevine tells me this will be *Manhunter: Chicago*, but that remains to be seen. I can heartily recommend this game to virtually anyone. The only exceptions are those who either detest, or for some reason cannot play, arcade games. The choice of interface makes it easy to play, the pace is quick, the plot enticing. Graphics are up to Sierra's standard. One last thank-you to Sierra for letting us use our hard disks, and please, do buy this game; don't steal it. Sierra has made a commitment to ST owners, let's not let them down.

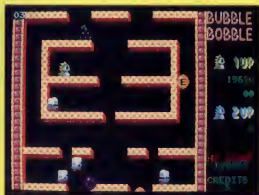


Manhunter: New York is intended for mature audiences. Parents should consider whether their younger, impressionable children should play this game.

Bubble Bobble

Taito
 267 West Esplanade
 North Vancouver, BC
 Canada V7M 1A5
 (604) 984-3344
 \$29.95, Color only

Reviewed
 by
 Clayton Walnum



In the past ten years, I doubt I've spent five dollars at an arcade. So when it comes to a program like *Bubble Bobble* from Taito, I can't tell you how it compares with the original. I can say, however, that it's a delightful game.

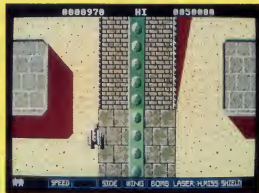
In *Bubble Bobble*, one or two players control (via joystick) Bub and Bob (two of the cutest brontosaurus you'll ever run into) throughout the 100 screens that make up their world. The two dinosaur friends have many enemies (I don't know why they're too darned cute to dislike), so they've developed a most unusual defense mechanism: they blow bubbles. These bubbles will trap within them any creature they contact, after which the bubbles will rise slowly to the top of the screen.

continued on page 94

Alcon ST

Taito
 267 West Esplanade
 North Vancouver, BC
 Canada V7M 1A5
 (604) 984-3344
 \$39.95, Color only

Reviewed
 by
 Frank Eva



Taito made its appearance in the ST software market with an excellent conversion of the coin-operated arcade game, *Arkanoid*. No doubt many ST owners were looking with eager anticipation to other releases from this giant of the industry. For many reasons, however, *Alcon* does not live up to expectations.

Alcon comes packaged in Taito's usual bifold slipcover and contains an historical scenario presented in comic-book style. It seems that the human inhabitants of the planet Theon, who fled here when their previous world, Orac, was attacked, are again under siege. ALCON, the Allied League of Cosmic Nations, must now attack the aliens in order

continued on page 94

Space Station Oblivion

Reviewed
by
Steve Panak

Epyx
600 Galveston Drive
P.O. Box 8020
Redwood City, CA 94063
(415) 366-0606
\$49.95, Color only



When I first receive most new games, I rarely

know what to expect just from looking at the box. Sure, Infocom's probably contains a work of interactive fiction, and SSI's an engaging simulation/adventure, but these are the exceptions. Another exception is any game that belongs to Epyx's Masters Collection. These games are sure to feature a challenge—one which, realistically, might never be surmounted. *Space Station Oblivion* provides just that.

The manual opens by setting up the scenario, although, as with most arcade games, the rationale for your struggles is best forgotten. Regardless, for those who are interested, it seems that your world has, through the lack of proper planning and rampant, uncontrolled development, accumulated a core danger-

continued on page 94

Technocop

Reviewed
by
Steve Panak

Epyx
600 Galveston Drive
P.O. Box 8020
Redwood City, CA 94063
(415) 366-0606
\$49.95, Color only



There aren't many games that provoke an

audible response of shock or surprise in a seasoned video-game veteran such as myself. I like to think that I've seen it all. And while this newest game from Epyx does not establish an entirely new theme or signal a radical departure from tradition, it does tread bravely into familiar territory—and takes no prisoners.

Basically two complete games in one, *Technocop* lets you live a high-speed police chase and criminal arrest set in the not-too-distant future. After booting up, the first phase of play has you propelling your VMAX Twin-Turbo Interceptor down a desolate highway, al-

continued on page 94

Downhill Challenge

Broderbund
17 Paul Drive
San Rafael, CA 94903-2101
(800) 527-6263
\$34.95, Color only

Reviewed
by
Scott Wasser



There you are, at the top of what certainly must be the highest mountain in the entire world. You're all bundled up in fluorescent-color clothes and wearing a pair of uncomfortable plastic boots with a couple of long fiberglass slats strapped to your feet.

But hey, who's complaining? Skiing is fun. At least that's what all your skiing buddies told you.

So you try not to think about how all the puffy clothing makes you feel like a helpless teddy bear. You struggle to stay upright as you inch your way to the edge of the mountain where the ski trail begins.

Ah, finally! You've made it! You gaze down the trail, which looks more like a sheer cliff than a gently descending slope, and try to pick a path with plenty of soft, forgiv-



Falcon

Spectrum Holobyte
2061 Challenger Drive
Alameda, CA 94501
(415) 522-3584
\$39.95, color only

Reviewed
by
Steve Panak



About the last thing we need for the ST is another flight simulator. With maybe a dozen already out there, each displaying a different degree of complexity and realism, from a simple outer-space arcade dogfight to a fully detailed carrier-based F-18 Hornet, every armchair pilot should be able to find his favorite set of wings. Still, after testing Falcon from Spec-

ing snow and as few bumps as possible.

Then, surprising even yourself, you somehow manage to summon up enough courage to nudge your skis over the precipice and hurtle down the mountain.

It's at that moment—the moment you reach the point of no return—you realize just how much trouble you're in. During all the bundling and buckling, nobody ever taught you anything about skiing.

That is precisely the biggest problem with *Downhill Challenge*, a program that simulates ski racing. The program provides you with everything you need to vicariously engage in four different types of ski racing—except the knowledge required to successfully compete.

It's a shame, because *Downhill Challenge* has some merit. For one thing, it's the only skiing simulation I'm aware of for the ST. For ski nuts who trek great distances, stand in long lines and pay big bucks to participate in the sport—*Downhill Challenge* is a pleasant way to enjoy the activity when there's no snow on the mountain or no time to get there.

As a simulation, it does a credible job of recreating the sport. After booting the program, you have the

option of choosing to practice one of four ski-racing events: Slalom, Giant Slalom, Downhill or Ski Jumping. Or, you can up to five other players can "register" to compete in all four events (The computer can be one of the competitors).

After going through the preliminaries, you're presented with the main game screen, which is divided into three parts. The largest part is a window, in which all the action takes place. The view is from above and behind the skier you'll control.

Framing that window on two sides is information about what is taking place on the course, such as the skier's speed, elapsed time and gates missed. In the lower left-hand corner of the screen is a small window that indicates the type of event being run or, in the case of the ski jump, shows a profile of the airborne skier.

Downhill Challenge captures the sensation of skiing about as well as can be expected of a computer simulation. Attribute that to nice graphics and good animation.

Because the screen scrolls both horizontally and vertically, perspective is realistic. Both the foreground and background change as the on-screen skier blasts down the moun-

tain. That helps create the feeling of being on a ski slope rather than a treadmill. Even the spills capture the jarring, disorienting feeling of the real thing.

On the other hand, having to use a joystick to control the skier is a constant reminder that this is a simulation. Although the controls are easy to master and the response is good, there's simply no way that jiggling a joystick can even remotely emulate the physical moves involved in real skiing. Using the keyboard to control the skier is even less satisfying.

The most disappointing thing about *Downhill Challenge* is the way it leaves you snowbound in the documentation department. The word "challenge" in the program's title could easily refer to the difficulty many players will have figuring out what's going on.

Most avid skiers know how important it is for beginners to start with a lesson from a knowledgeable instructor. Smart skiers tout the value of taking a lesson every once in a while, no matter how long you've been skiing. The more you know about skiing, the more you'll enjoy the sport. But information is hard to come by in *Downhill Challenge's* user's guide. Only the

most rudimentary instructions are provided, and some important information is omitted.

For example, nowhere in the guide does it explain the controls required to compete in the ski jump event. It's up to the player to figure out how to control the skier.

A less glaring omission is the documentation's failure to explain the strange beeping you'll hear at certain times during a run, or that the "competition" mode requires players to participate in all four events and take two runs in each.

Finally, it disturbs me that the guide doesn't provide any explanation of the individual events or the differences between them. Ski racing is an esoteric sport. A simulation of it should be accompanied by documentation that includes paragraphs devoted to explaining the events and describing the techniques and skills required to master them.

The skimpy documentation makes *Downhill Challenge* less enjoyable than it could be. If you're familiar with ski racing or patient enough to figure out *Downhill Challenge*, you'll probably think it does a good job of simulating the sport. If only it did a better job describing it. ■

trum Holobyte, I get the feeling that there is room for at least one more.

This is probably due as much to the attention paid to teaching you to operate the complex simulator as to the program itself. I've played a lot of flight simulators, and it often takes me hours just to get off the runway, much less actually complete a mission. Setting flaps, revving up, even an operation as simple as starting the engines quickly becomes lost in the confusing collection of keyboard commands. Most packages solve the problem of what key performs a given operation by providing a reference card, but that still leaves open the issue of when to press it. *Falcon's* superb manual is designed to alleviate this problem as well.

Just thumbing through the pages will impress even the most skeptical user. The manual's 130-plus generously illustrated pages tell you not only everything you need to know about the program, but also

everything you'd ever want to know about flying an F-16. The first section contains a step-by-step tutorial on taking off—a welcome sight for me. With the manual on my lap and a finger always ready to hit the pause key, I managed to leave the ground on my first try. Additional sections detail each of the many controls and missions contained in the game, as well as advanced fighter tactics. A perfect score of ten for this packaging.

In addition, one of the most interesting aspects of *Falcon* is the unique way in which it approaches the problem of copy protection. Fortunately, like most sophisticated ST games, it eschews a crude disk-protection scheme, opting instead to rely on the entry of a password from the documentation. But rather than requesting the input of a word from an easily photocopied manual, *Falcon* requests the input of a password from a window in a paper dial, which is twisted to line

up two symbols displayed on the screen. This dial, similar to that used to convert inches to centimeters, is a difficult item to copy. As long as copy protection is necessary (and it is), this is the best way to go. Just make the symbols a little more distinct next time.

Getting back to the game itself, once you load up, you choose your rank, from First Lieutenant to Colonel, which in turn determines difficulty. Starting at the lowest difficulty level takes some of the sting out of crash landings, while the upper levels provide all the realism of actual combat missions. For instance, *Falcon* is the first program I've seen that attempts to simulate the blackout that occurs when a fighter jet turns too abruptly. This builds upon the usual difficulty modifiers, such as limited fuel and armaments. As these resources become depleted, it will be up to you to select the proper mix for a successful mission. The package con-

tains 12 missions, which range in difficulty from a simple bombing run to complex dogfights.

The display is standard stuff. Your main view is out of the front of your canopy and contains most of your gauges in addition to the heads-up display, which superimposes the most important readouts onto the canopy itself. Left and right cockpit views access additional gauges and warning lights. In contrast to the superb graphics that form the cockpit, the landscape is a disappointing but adequate vector-graphic view. Additional features include a black box that records your flight for later analysis and has the ability to play head-to-head (using another computer) via modem.

I don't know what more to tell you about *Falcon*. If you don't already have a flight simulator, you should seriously consider making *Falcon* your simulator, as no library is complete without one, and *Falcon* is most definitely flightworthy. ■

Bubble Bobble continued

Obviously, a bubble is an insecure prison for a ferocious creature. If you (as Bub or Bob) are to get rid of the bubble's victim permanently, you must pop the bubble before it bursts on its own. If you wait too long, the angry occupant will escape and zip around the screen at lightning speed.

When you pop an enemy-laden bubble, the creature it contains changes into a piece of fruit (don't ask me to explain it), which may be scooped up for extra points.

Sometimes special targets appear, which, when run over, give bonus points. Also, special bubbles containing fire, thunder or water, if popped, give additional help in clearing the screen. For example, a water bubble releases a flood on which the dinosaurs can ride, destroying everything in their path. The lightning bubble releases a bolt of lightning that streaks across the screen, blasting all it touches.

Other bubbles that may appear contain letters. If you gather enough letters to spell "extend," you get an additional life and move on to the next level.

Bubble Bobble's graphics are adequate, though not sensational. Each screen is constructed of various floors that Bub and Bob, as well as the other creatures, can jump between. There is a minimum of animation, though enough to keep the characters interesting. Perhaps the best animation is the way Bub and Bob waddle about, wagging their tails. It's difficult not to smile.

As is typical with Taito's games, high scores are not saved to disk. One has to wonder why they bother having a high-score screen at all. Why take the time to enter your initials when they'll be gone the next time you boot the game? This is only a minor defect, of course, but I can't imagine why Taito doesn't go to the extra effort of including this feature in their games. It would be easy to add.

Every once in a while someone designs a program that takes an old standard and cleverly enhances it, creating something original and fresh. *Bubble Bobble*, although basically a shoot-'em-up, has transcended its genre and is unique and playable. Recommended. ■

Alcon ST continued

to liberate their planet and the hostages held there.

The player pilots the most advanced starfighter, the SW475, capable of firing forward-projecting lasers, homing missiles and bombs, as well as side-firing energy weapons. Unfortunately, only one player at a time can play *Alcon*. Not only is there no simultaneous two-player mode, there isn't even an alternating two-player mode.

If the reader is not familiar with the *Alcon* coin-op, perhaps *Xevious* rings a bell. The two games are similar, being basically vertically scrolling shoot-'em-ups. The only difference in play mechanics is the way bombs are handled. I believe *Xevious* allowed a player to bomb stationary targets in a realistic bombing scenario. In *Alcon*, bombs are similar to forward-firing lasers in that they explode a certain distance in front of the craft, not below it.

In a nutshell, what the ST owner gets with *Alcon* is not a direct conversion from the original coin-op, but a pale imitation. The background graphics look like they came from an unexpanded IBM PC, although this is probably not possible since an IBM version hasn't been released.

In addition to the lackluster background graphics, *Alcon* features unartistically portrayed attackers who have little or no animation. Further, the shadow of your ship constantly flickers; and even though there are usually fewer than six enemies on the screen at one time, the program noticeably bogs down in its execution. The SW475 at times moves painfully slow for an arcade game.

Also, the "better" weapons systems don't add much capability to the ship. The lasers are almost useless, since they have such a limited range. The shots won't even go far enough to make it off the screen unless you're flying within the top half of the action area. The wing guns don't add firepower; they just improve the range of the lasers while at the same time making the ship larger and thus easier to hit. The side-firing energy weapons are more like projecting shields—only if they happen to collide with something are they effective. This

means you have to be close to your enemy to use them.

One last note: The high scores cannot be saved to disk. Even though this was also the case with *Arkanoïd*, that game was so good that the lack of a high-score save could be forgiven.

Upcoming releases from Taito, *Operation Wolf* and *Renegade*, will use the actual arcade-machine graphic files for the ST conversions. Hopefully, these will take advantage of the ST's capabilities. *Alcon* sure doesn't. ■

Space Station continued

ously full of natural gases. Your mission is to place a drilling rig in each of the 18 platformed sectors and safely vent the trapped gas into the atmosphere. Thus, the game is a race against time to first locate each sector, then identify the proper place to drill. Complicating your life are a limited energy reserve and a whole slew of security devices.

Your craft is an excavation probe, which derives its power from the energy stored in the Rubicon crystals strewn about the planet. Needless to say, one main strategy is to find and secure an ample supply of these crystals, buying yourself enough time to complete the game. But not too much time, as you have only four Mitral hours to complete the mission. Moving, firing your laser or drilling for gas uses up energy, while numerous tricks and traps must be solved or avoided to gain access to necessary areas. For instance, switches, when hit by a bolt from your beam laser, open doors and dissolve energy fields. Hidden teleporters transport you to unexplored sectors, and hidden somewhere on the planet is a reconnaissance jet. Locating and flying this craft will allow you to chart Mitral's many areas. The sheer variety of the conflicts is likely to keep most players engaged for months.

As with most ST arcade-style games, *Oblivion's* graphics shine. Although the blocklike vector graphics that outline Mitral's features are standard stuff, the image of the ship's control panel is dazzling. The ship seems so real you can almost

feel the burnished steel of the command console, almost sense the vibrations emanating from its power plant.

You control the game either with the keyboard or the mouse (the mouse is recommended). Moving the pointer across the command console accesses such options as game save, as well as allowing movement of the probe and positioning of your drilling rigs. Detailed readouts and indicators keep you apprised of every crisis. Thoughtful design and programming make control simple and intuitive, allowing even the new player to direct his full attention to the formidable task of completing his mission.

The manual concisely explains every feature of the game, but provides few hints. For those who are unable to complete the game, Epyx has a hint book that includes not only *Oblivion*, but also *Sub Battle*, *L.A. Crackdown* and *The Legend of Blacksilver*.

Space Station Oblivion gets good marks for its overall challenge and smooth execution. The only drawback is the lack of detail in the planet's graphics, which compare poorly to the high detail of the command interface. *Space Station Oblivion* is guaranteed to occupy your disk drive, and your time, for months to come. ■

Technocop continued

ways on the lookout for a felon. Once alerted via your on-board computer, you race to the scene of the crime, blasting or bumping DOA (Death On Arrival) criminals off the road or breezing by them at more than 200 mph, keeping in mind that within each vehicle are your enemies. If you survive and arrive before the clock runs out, you'll progress to the second stage of play.

Exiting your car, you enter the tenement building within which hides the crime kingpin you seek. Hiding out with him is his band of ruthless henchmen, who will stop at nothing to protect their boss. Successfully making it through all the tenements requires reflexes that would turn Dirty Harry green with envy. Every gamer without an ST

will turn green when they see these graphics.

When you're in your car, the screen display is predictable enough: an instrument console beneath the view from above and behind your car. But the first time you exit the vehicle and enter a crime scene, I personally guarantee that there's no way you'll be ready for what you'll see. The bottom of the display fills with an image of your left arm, upon which is strapped your wrist command console. This futuristic device displays a mug shot of your adversary, along with your orders: whether he is to be brought in dead or alive. The midsection of the console contains radar that shows the location of the lawbreaker, crime clock counting down your time limit, and health and strength indicators. Large flashing buttons indicate whether you have chosen a gun or a net as your weapon.

As you walk toward the crime boss, the scenery that scrolls smoothly by is as ugly as the world you have chosen to inhabit. Graffiti adorns the walls of the slum, innocent bystanders wander the halls. Just don't get lulled by the great graphics, for the moment you least expect it, a punk will appear and try to waste you. You're particularly vulnerable as you leave an elevator or walk by a door, which may suddenly open with a realistic creak, revealing a deadly delinquent.

Your biggest shock will come when you blow away the first minor hood. Suffice it to say that there is little left of the punk's twitching body after taking a few slugs from your .88 magnum. While this graphic display might be worthy of an "R" rating in a motion picture, this cartoonish mayhem is just good fun.

If you survive his goons, you'll discover that each crime chief has his own *modus operandi* and must be dealt with in a specific way. In addition, each tenement is a maze that must be mastered before progressing to the next. All these factors make for a game as addictive as any I've seen.

The game is not perfect. Close, but not quite. The main problem involves the lengthy disk swaps and loads that accompany the

transitions between the two phases of play. Each portion of the game, being a nearly complete game in and of itself, seems to occupy all the ST's memory. To continue nitpicking, I thought the driving portion could have been a little stronger, maybe more like *Spyhunter*, with a greater variety of weapons and scenery. But despite these misgivings, I can't think of anyone who wouldn't be satisfied with *Technocop*. It would be a crime to leave it on your dealer's shelf. ■

Steve Panak has written more game reviews for ST-LOG and ANALOG Computing than anyone on the planet. He lives in Ohio, where he plays games on his ST and, with the time remaining, practices law.

Frank Eva is an auditor by profession, but has been involved in the computer industry ever since his purchase of an Atari 400 many years ago. He has dabbled in programming and has had several text adventures published.

Scott Wassser has been a daily newspaper reporter and editor for the past 12 years, and has been interfacing with computers for the last four. He has written columns and feature stories about computer hardware, software and home electronics, and is a regular reviewer for ST-LOG.

Clayton Walnum is the editor of ST-LOG and ANALOG Computing.

Peter Smith lives in Sag Harbor, NY, with his in-house editor, Julie. When not in front of his computer, he is a bicycle enthusiast, a struggling musician and a poet. He can be reached on GEnie at the address CaptCook.



ULTIMATE STORAGE

Here's the perfect way to organize your **ST-Log** library—sturdy, custom-made binders and files in burgundy leatherette with embossed gold lettering. Gold labels are included to index by volume and year. One binder or a box-style file is all you'll need to accommodate 12 issues (1 year) of **ST-Log**—all the games, programs, tutorials and utilities that you want handy.



The **ST-Log** binder opens flat for easy reading and reference. They're economically priced at only \$9.95 each—3 binder for \$27.95 or 6 binders for \$52.95.

The **ST-Log** file is attractive and compact, holding 12 issues for easy access. Files are available for only \$7.95 each—3 files for \$21.95 or 6 files for \$39.95.

Add \$1.00 per case/binder for postage and handling.
Outside U.S., add \$2.50 per case/binder—U.S. funds only.

I enclose my check or money order in the amount of \$_____

Send me: _____ **ST-Log** files
_____ **ST-Log** binders.

PLEASE PRINT.

Name: _____

Address: (No P.O. Boxes) _____

City: _____

State: _____ Zip Code: _____

Send your order to:

Jesse Jones Industries

DEPT. ACOM, 499 East Erie Ave.,
Philadelphia, PA 19134

Call Toll Free 1-800-972-5858

7 days, 24 hours.

Charge orders only, minimum \$15.00

PA residents, add 6% sales tax

Satisfaction guaranteed or money refunded.

FROM THE MINDS OF THE CODEHEADS COMES:



- Run any ST program simply by pressing a "hot" key...from the ST desktop!
- No more wading through folders to run programs.
- Up to 54 programs may be installed. **\$39.95**
- Load and Save complete sets of programs.
- Run a program by clicking on its name in the HotWire menu, or by pressing its "hot" key from the menu or desktop!
- Allows full use of ST desktop features.
- Unique "work file" command line features make HotWire an excellent shell for developers.

MIDIMAX MIDI software for the ST

- The Real-time music performance aid!
- Create strings of MIDI commands triggered by any MIDI event.
- Sound like an entire section of instruments all by yourself!
- Real-time multi-voice, multi-channel harmonization.
- Single notes can produce chords to 18 notes on any channel(s).
- Instant switching between 8 MIDI chord maps.
- Turns your ST into an intelligent THRU box.
- Unlimited keyboard splitting, filtering, and remapping.

\$49.95

MULTI-DESK UNLIMITED DESK ACCESSORY POWER FOR THE ATARI ST

- Load and use up to 32 accessories or more at any time...without rebooting! **\$29.95**

G+plus THE POWERFUL AND COMPLETE REPLACEMENT FOR GDOS

- Join the ranks of users who are free of GDOS slowdown.
- Automatically load the correct ASSIGN files for each program you use. **\$34.95**

Phone: (213) 366-5735 CodeHead Software
 Visa and Mastercard accepted. P.O. Box 74090
 Shipping charge: US \$2, Canada \$5, Europe \$5. CA Los Angeles, CA 90004
 residents add 6.5% sales tax. (We're 100% assembly language!)



CIRCLE #107 ON READER SERVICE CARD.

RENTING SOFTWARE ISN'T HARD!

It's as easy as picking up the phone and giving your order. If you have a credit card, it's even easier. The hardest part may be waiting for the mail to come!

We're having a special sale, with up to 80% off selected software. Call now for a complete list.

Call toll-free outside Texas: 1-800-433-2938
 — Inside Texas call: 817-292-7396

WEDGWOOD RENTAL
 5316 Woodway Drive
 Fort Worth, Texas 76133



CIRCLE #108 ON READER SERVICE CARD.

The ST-LOG #35 diskette contains 24 magazine files. They are listed below.

FILENAME.EXT	FILE TYPE	COMMENTS
CHANSHP.ARC contains:		
CSS .C	C	C-HANSHP
POPARR.ARC contains:		
ADDRESS .PRG	RUN FILE	ADDRESS BASE
ADDRESS .TXT	TEXT	INSTRUCTIONS
DOT-2-DOTS.ARC contains:		
DOT2DOTS.PRG	RUN FILE	DOT-TO-DOTS
DOT2DOTS.DAT	DATA FILE	DATA FILE
DOT2DOTS.LST	GFA BASIC	SOURCE CODE
DOTMAKER.PRG	RUN FILE	PUZZLE MAKER
DOTMAKER.LST	GFA BASIC	SOURCE CODE
EXPGR42.ARC contains:		
EXPGR42 .GFA	GFA BASIC	EXPERIMENT 42
GFAASRO.PRG	RUN FILE	GFA RUN ONLY
MAGNITER.ARC contains:		
MAGNITER.PRG	RUN FILE	MAGNIFIER
MAGNITER.TXT	TEXT	INSTRUCTIONS
HELP .TXT	TEXT	INSTRUCTIONS
MACCODES.TXT	TEXT	MACCODES
QUICKREP .TXT	TEXT	QUICKREP
REAME .DOC	TEXT	REAME
TEMPLE.TXT	TEXT	TEMPLE
XYZZZ .TXT	TEXT	XYZZZ
CONFIG .TOS	RUN FILE	CONFIG
CONFIG .TXT	TEXT	CONFIG
MANSWEN.ARC contains:		
MANSWEN.PRG	RUN FILE	MANSWEN MONITOR
SAVER.ARC contains:		
SAVER .PRG	RUN FILE	SCREEN SAVER
SAVEREN.S	ASSEMBLY	SOURCE CODE
ARCV .TTP	RUN FILE	UNARCHIVING PROGRAM
REAME .DOC	TEXT	Disk Instructions
UNARCHV.DOC	TEXT	Unarchiving Instructions
Disk Instructions:		
Only those files with .PRG, .TOS, or .TTP extensions may be run from the GEN desktop.		
Other programs may require additional software as shown below. The files on this disk have been archived (compressed) into .ZIP files. To restore the programs to their runnable form, follow the instructions found in the UNARCHV.DOC file.		
NOTE! Due to space limitations, the source code for MANSWEN Monitor is not on this disk. If you wish to obtain this file, send your original September '89 disk (after making a copy) the original will be erased, along with a postage-paid mailer to ST-LOG SEPTEMBER SOURCE CODE P.O. Box 1413, MANCHESTER, CT 06040-1413		



.EXT	DESCRIPTION
.BAS	Requires GFA BASIC
.LST	Requires GFA BASIC
.C	Requires C compiler
.S	Requires 68000 assembler
.PAS	Requires Pascal compiler

WARNING: Be sure to read the appropriate magazine articles before attempting to run the programs on this disk. Failure to do so may yield confusing results.

INDEX TO ADVERTISERS

ADVERTISER:	PAGE:	READER SERVICE #
AVANT GARDE	39	104
BRE SOFTWARE	27	102
CODEHEAD SOFTWARE	96	107
DIGITAL VISION	87	106
INTELLIGENT MUSIC	73	105
ISD MARKETING	Cover 2	101
MASTERTRONIC	Cover 4	111
MICROTYME	31	103
MIGRAPH	Cover 3	110
PROCO PRODUCTS	98	109
WEDGEWOOD RENTAL	96	108

The index is an additional service. While every effort is made to provide a complete and accurate listing, the publisher cannot be responsible for inadvertent errors.

Keeping Up

KARL E. WIEGERS

If you're serious about keeping up with happenings in the computer world, you've probably made the same discovery I have: It's almost impossible. The rate at which new information about computers is being generated is astronomical, as evidenced by the wide variety of magazines on the newsstand. Unless you're a lottery winner with 24 leisure hours per day, you can't read them all. So what's an enthusiast to do?

I regularly peruse about a dozen periodicals in a feeble attempt to keep up with my profession. Many of these also would be interesting to the serious hobbyist. None of these magazines is hardware-specific, although the words "IBM compatible" crop up a lot. I assume you already read STLOG. The world also is filled with magazines devoted to MS-DOS computers and machines named for fruit, but I'll ignore those for now. Instead, I'll describe some general computer publications you may not know about.

Don't be surprised if you don't see the word "Atari" often in these publications. The American computer world barely knows we exist, although Jerry Pournelle, who writes an entertaining column in *BYTE*, puts in a good word for Atari every now and then.

BYTE is a leading microcomputer magazine containing a wealth of information. It's a monthly, with a subscription price of \$22.95 per year. Recent issues weigh in at a hefty 400 pages or so. Regular features include product announcements for all sorts of micros, short handson reviews of new products, preliminary assessments of significant machines, thorough analyses of exciting new hardware and software, book reviews,

and advice columns from an assortment of experts in various fields: applications, Macintosh, OS/2, communications and business computing.

Each issue has an in-depth containing several articles on different aspects of a major theme in contemporary computing. In addition, several feature articles on technology, software or human factors are included. Finally, hands-on columns deal with do-it-yourself hardware and software projects; a recent series described a homebuilt supercomputer! *BYTE* is an excellent publication, well worth buying if you can store the hefty volumes on a reinforced floor.

Another good monthly is *Personal Computing*, subtitled "The Personal Systems Magazine." A subscription is \$18 per year, and the most recent issue I saw had 320 pages. In addition to an assortment of feature articles on contemporary personal computing issues (MS-DOS vs. OS/2 operating systems, online searching tips from professionals), there are columns on connectivity, communications, technology, product trends, desktop publishing, databases, the computer industry, and so on. Reviews and buyer's guides are prevalent, as are new product blurbs.

Computerworld is a weekly newspaper averaging about 120 pages per issue; a

one-year subscription is \$44. It covers news on all sizes and shapes of computers and computer companies, with particular emphasis on business and technology moves by vendors. There are regular sections on systems and software, microcomputing, networking, management, the computer industry, computer careers, training, industry trends, executive input, opinion and analysis. The focus is mostly on corporate computing, so product reviews aren't common. If you're a computer professional seeking a new job, check out the classified ads in the back.

Software Magazine is free to data-processing professionals who meet certain minimum criteria. (I qualify, so you know they're minimum criteria.) If you're not a lucky winner, it will cost you \$40 a year for 15 issues of 80-90 pages each. This is a very readable publication, probably because it's written "for managers of corporate software resources." Much of it is interesting to normal people, too.

Software Magazine includes current market and corporate computing information, discussing strategies of major hardware and software vendors. Current trends in computer-aided software engineering (CASE), databases and other technical areas are revealed. These arti-



cles often include comprehensive tables of current products (such as CASE tools) from different vendors, along with a reader inquiry card to help you get more information. Hot new products for the mainframe, mini and micro worlds are mentioned too.

If you're a serious programmer type, you should check out *Dr. Dobbs' Journal of Software Tools*. For about \$30 you can get a year's worth of monthly issues that average 150 pages of meaty articles. *Dr. Dobbs' Journal* has columns on C programming, structured programming, programming paradigms (whatever those are), Forth, programming the Macintosh, and artificial intelligence. Feature articles address topics in many languages, including assembly, Ada, C, Lisp, Forth, Prolog, Modula-2, SQL and BASIC. You'll also find articles on Unix, DOS, programming utilities and tools and contemporary programming technologies. Quite a few program listings are included, but don't expect anything to run on an Atari. The advertisements include pitches for all sorts of compilers, languages, and programming tools and aids.

A bit more technical, *Computer* is the monthly magazine published by the IEEE Computer Society. IEEE is an international engineering association, and they kind-

ly let software engineers join now, too. Membership in the IEEE Computer Society is \$39 per year, which includes a subscription to *Computer*. Membership requires a certain level of formal technical education; I guess they let me in because I spent the 1970s learning how to be a chemist.

Each 140-page issue of *Computer* contains several technical articles on contemporary issues in computing, which are sophisticated but not completely impossible to follow (up to a point). Regular departments cover standards in computing (IEEE publishes many engineering standards), upcoming conferences, book reviews and new-product reviews (usually theme-oriented, with quite a bit of MS-DOS software and hardware). If you're contemplating a new career, classified ads for computer positions in the academic world are included.

Another IEEE publication is *IEEE Software*, which appears bimonthly. You must be a member of IEEE to subscribe, but you could probably find it at college libraries. This is just as technical as *Computer*, but it focuses on software (surprise!). Theme articles, like a recent one on object-oriented programming, are useful condensations of important software topics. Overall, I find the articles quite

readable. My favorite sections are departments on human factors, software quality standards and legal aspects of software. I've obtained many useful tips from the human factors and quality columns.

IEEE Transactions on Software Engineering, published monthly by the IEEE Computer Society, contains more serious stuff. Each 200-page issue contains academic papers on hard-core research in software engineering. This is a bona fide scientific journal, not for the faint of heart. Some recent papers discussed performance evaluation, concurrent programming, distributed processing, exploratory system analysis tools and sophisticated database topics. These papers are written by researchers at universities, IBM, Bell Laboratories and the like. I'm lucky to get past the first page of most of the articles, but often I don't even understand the titles.

These eight periodicals are by no means all that are available, but they should be enough to help you keep up with the computer world. Let's see how much of your valuable time it might take to read these babies religiously. Adding up the number of printed pages each month in these eight publications, I come up with a total of about 1,800. Let's assume that only half of these pages contain useful information, the rest being ads, pictures and whatnot. This gets you down to some 900 pages per month. An average magazine page contains around a thousand words, so I estimate 900,000 words per month. If you can read 300 words per minute, which is pretty good for technical stuff, you'll spend around 50 hours a month reading these magazines! Not very practical.

So, how can you keep up with computer literature? You can't. Pick out the most interesting articles and forget the rest. It seems a shame to ignore most of each issue, but sometimes you have to do other things, like eat, sleep and use your Atari. ■



◀ **Karl E. Wieggers**, a longtime contributor to both *ST-LOG* and its sister magazine, *ANALOG Computing*, is a software engineer in the Eastman Kodak Photography Research Laboratories. He lives in Rochester, New York. *ST-LOG* invites all authors to submit essays for possible use in the Footnotes department. Submissions should be between 1,000 and 1,500 words and may be on any aspect of Atari computing. Any style or type of essay is acceptable—opinion, humor, personal experience—but creativity is a plus. Send your submission to: Footnotes, c/o *ST-LOG*, P.O. Box 1413-M.O., Manchester, CT 06040-1413.

ProCopy ST BACKUP UTILITY

You can't backup your software because copy protection locks you out. **ProCopy** is the key!

- Protects against the accidental loss of expensive software
- Works with all Atari STs
- Copies both S/S & D/S disks
- Use with 1 or 2 disk drives
- Not copy protected
- FREE shipping and handling
- TOLL-FREE telephone number
- Updates available to registered owners
- Orders shipped same day
- Disk analyze function included



and C.O.D. orders

Call (800) 843-1223

Send check for \$34.95 (overseas add \$2.00 for air mail) to:

PROCO PRODUCTS

P.O. Box 665, CHEPACHET, RHODE ISLAND 02814 USA
(401) 568-8459

Available in Europe

THE MOVING FINGER CO.
Building 2
Sharnock Quay
Southampton, SO1-1QL
England
Tel. 0703-229041

TECH-SOFT
COMPUTER WHOLESALE
324 Stirling Highway
Claremont
Western Australia 61010
Tel. 09-385-1885

CIRCLE #109 ON READER SERVICE CARD.

Dealer
Inquiries
Welcome

\$ 34.95

Enhance your publishing power.

Migraph Hand Scanner.



Add scanned graphics to all your projects quickly and easily with the Migraph Hand Scanner.

The Hand Scanner provides a scanning window over 4" wide with four scanning resolutions: 100, 200, and true 300 and 400 dots per inch. It also offers adjustable contrast, three dither settings for scanning halftones, and a special setting for line art.

Build your own library of images. Logos, photographs, books, illustrations . . . provide you with an unlimited supply of art.

Touch-Up,[™] Migraph's high resolution image editor, lets you scan and edit your images. Powerful editing tools include crop, invert, resize, flip, cut and paste, multiple zoom levels, a full array of paint tools, and much more.

When your images are "pixel perfect", import them into your favorite ST publishing programs like Calamus, Fleet Street, PageStream, and Publisher ST. In addition, several

save formats let you use your images on the PC and Mac.

The Migraph Hand Scanner and Touch-Up. Powerful tools for professional publishing.

For all Atari ST and Mega systems with monochrome or color monitors, 1MB memory and 3.5" DS disk drive.

See your dealer today for more details or call us direct. Dealer inquiries invited.



200 S. 333rd St., Suite 220

Federal Way, WA 98003

(800) 223-3729

(206) 838-4677

© Copyright 1989 Migraph, Inc. The Migraph logo is a registered trademark and Touch-Up is a trademark of Migraph, Inc.

CIRCLE #110 ON READER SERVICE CARD.

THEY'RE BACK!



ON A COMPUTER NEAR YOU

	IBM	APPLE II	MAC	C64	AMIGA	ATARI ST
SCRABBLE®	\$39.99	\$39.99	\$39.99	\$29.99	\$39.99	\$39.99
MONOPOLY®	\$39.99	\$39.99	DEC	\$29.99	\$39.99	\$39.99
RISK™	\$39.99	\$39.99	\$39.99	\$29.99	\$39.99	\$39.99
CLUE® MASTER DETECTIVE™	\$39.99	\$39.99		\$29.99	\$39.99	\$39.99

Monopoly® © 1935, 1985. Produced under license from Parker Brothers, division of Tonka Corporation. Risk™ © 1989, 1989. Used under license from Parker Brothers.
 Clue® Master Detective™ © 1988. Produced under license from Parker Brothers, division of Tonka Corporation, and Waddington Games Limited.
 Scrabble® Licensed by Selchow and Righter Company, a subsidiary of Coleco Industries, Inc. Scrabble board design © 1948 Selchow & Righter Company.



Distributed by: VIRGIN MASTERTRONIC INTERNATIONAL Inc.
 18001 Cowan, Irvine, California 92714 U.S.A.
 Telephone (714) 631-1001 Telefax (714) 631-1004

CIRCLE #111 ON READER SERVICE CARD.

